

**THE CATHOLIC UNIVERSITY OF AMERICA
DEPARTMENT OF ELECTRICAL ENGINEERING**

**FINAL REPORT
on
DEVELOPMENT OF ADVANCED CONTROL SCHEMES
FOR TELEROBOT MANIPULATORS**

Research Grant NAG 5-1124

Charles C. Nguyen
Principal Investigator and Associate Professor
and
Zhen-Lei Zhou
Graduate Research Assistant

submitted to
Mr. Gary E. Mosier
Code 712.1
Goddard Space Flight Center (NASA)
Greenbelt, Maryland

February 1991

REPORT SUMMARY

This is a final report presenting the research results obtained from the research grant entitled "Development of Advanced Control Schemes for Telerobot Manipulators," funded by the Goddard Space Flight Center (NASA) under a research grant with Grant Number NAG 5-1124, for the period between February 15, 1989 to Dec 31, 1990.

To study space applications of telerobotics, Goddard Space Flight Center (NASA) has recently built a testbed composed mainly of a pair of redundant slave arms having 7 degrees of freedom and a master hand controller system. This final report presents the mathematical developments required for the computer simulation study and motion control of the slave arms. The first part of the report presents the slave arm forward kinematic transformation which is derived using the D-H notation and is then reduced to its most simplified form suitable for real-time control applications. The vector cross product method is then applied to obtain the slave arm Jacobian matrix. Using the developed forward kinematic transformation and quaternion representation of the slave arm end-effector orientation, computer simulation is conducted to evaluate the efficiency of the Jacobian in converting joint velocities into Cartesian velocities and to investigate the accuracy of the Jacobian pseudo-inverse for various sampling times. In addition, the equivalence between Cartesian velocities and quaternion is also verified using computer simulation. In the second part of the report, we deal with the motion control of the slave arm. Three control schemes, the joint-space adaptive control scheme, the Cartesian adaptive control scheme and the hybrid position/force control scheme are proposed for controlling the motion of the slave arm end-effector. Development of the Cartesian adaptive control scheme is presented and some preliminary results of the remaining proposed control schemes are presented and discussed.

1 Introduction

Kinematically redundant¹ manipulators classified as manipulators whose number of degrees of freedom (**DOF**) is greater than that of task space coordinates has been subject of considerable research in the last several years [1, 8] because of their many advantages as compared to non-redundant manipulators. In a non-redundant manipulator, there exists a finite set of joint variables and associated manipulator configurations such as elbow up or elbow down for a given position and orientation of the manipulator end-effector. Thus the manipulator joint motion is uniquely determined for a prescribed end-effector trajectory and a given pose. Consequently, non-redundant manipulators are limited in their ability to track an arbitrary end-effector path because of singularities, joint limits, and obstacles, which might occur along the corresponding joint trajectories. On the other hand, in redundant manipulators, a prescribed end-effector trajectory corresponds to an infinite number of joint motions due the redundant DOF's which enable the manipulator to avoid singularities and obstacles, to keep the joint variables within their physical limitations, to minimize kinetic energy and to provide greater dexterity. Recognizing the above advantages, robot designers have adopted redundant manipulators for future space robots which will replace or assist astronauts in performing space operations. A Flight Telerobotic Servicer (**FTS**) which is responsible for numerous tasks on the future NASA space station, such as assembly, inspection, servicing, and maintenance is currently under intensive study and development at the Goddard Space Flight Center (**GSFC**). An integral part of the research facilities at GSFC is a dual-arm telerobot system which consists mainly of a pair of 6-DOF mini-master controllers and a pair of 7-DOF redundant slave arms. The telerobot system serves as a testbed for investigating a variety of research issues of telerobotic operations in space, including zero-g operation, teleoperated and autonomous control, dual-arm manipulators, advanced control of redundant manipulators, hierarchical control etc. [9].

This report presents some mathematical developments which will be used in the computer simulation study and real-time control of the slave arm motion. In particular, we will focus on the manipulator forward kinematics, differential motion analysis and propose three control schemes for the slave arms. The organization of this report is described as follows. Next section will give an overview of the GSFC telerobot system and briefly describe the structure of the slave arm. Then the forward kinematic transformation for the manipulator is derived in its most simplified form using the Denavit-Hartenberg notation. After that, we obtain the manipulator Jacobian using the vector cross product method and then discuss the pseudo-inverse of the Jacobian. Computer simulation is then conducted to evaluate the efficiency of the Jacobian in converting joint velocities into Cartesian velocities, to investigate the accuracy of the Jacobian pseudo-inverse for various sampling times and to verify the equivalence between Cartesian velocities and quaternion. Finally three control schemes, the joint-space adaptive control scheme, the Cartesian adaptive control scheme and the hybrid position/force control scheme are proposed for controlling

¹The term "redundant" is often used instead of "kinematically redundant".

the motion of the slave arm end-effector. Development of the Cartesian adaptive control scheme is presented and some preliminary results of the remaining proposed control schemes are presented and discussed.

2 The GSFC Telerobot Testbed

The GSFC Telerobot Testbed as shown in Figure 1 is composed mainly of a *master arm* system and a *slave arm* system and interfacing/control devices. The master arm system, the Kraft Mini Master (**KMM**), manufactured by Kraft Telerobotics, Inc., has a left arm and a right arm, each of which consists of a KMC 9100F-MC Force Feedback Master Controller and a KMC 9100-S Master Control Electronics System. Each master arm has 6 DOF's arranged to provide two assemblies, a shoulder assembly to provide the primary motions of azimuth, shoulder elevation and elbow, and a wrist assembly to provide roll, pitch and yaw. Position feedback is obtained through potentiometers mounted on the six joints. The slave arm system, manufactured by Robotics Research Corporation (**RRC**) consists of a pair of K-1607 slave arms, each of which is an anthropomorphic redundant manipulator having 7 DOF's with human-arm-like tool-handling dexterity. The arm mechanism is a series of joint drive modules, each of which contains an electric servomotor, harmonic drive gear reducer, joint position and torque transducer, joint travel limits and associated structural elements. GSFC engineers have studied the RRC controllers and some hardware modifications were performed to accommodate future implementation of advanced control schemes such as adaptive and intelligent control, and other advanced features such as high-speed parallel processing.

We now use Figure 2 to explain the operations of the telerobot system. As the figure shows, force sensors and joint position/velocity sensors mounted on the slave arms provide the RRC slave arm system with feedback data of joint forces and joint positions/velocities, respectively. Force reflection at the KMM system can be achieved by applying an appropriate coordinate transformation on the slave arm joint forces. A task can be performed either in a *teleoperated mode* or *autonomous mode*. In the teleoperated mode, the human operator residing in an operator control station remotely controls the motion of the slave arms via the KMM arms using familiar hand and arm movements while observing the slave arm motion and the task space from a window or a TV monitor. The force/torque applied by the human operator on the KMM handles produces 6 joint forces/positions in the master arm system, which then are converted to 7 corresponding joint forces/positions in the RRC slave arm system via an appropriate coordinate transformation. The 7 joint variables will then serve as the reference inputs to the control system of the RRC arm. Based on the errors between the reference inputs and actual joint variables provided by feedback data, and governed by a control scheme, the controller sends appropriate signals to the slave arm actuators so that the end-effector tracks the desired motion with minimum tracking errors and simultaneously applies a desired contact force on the task environment. In addition, the human operator can feel the forces exerted on the end-effector by means of a force reflecting system which produces back-driving forces in the

master arm joint actuators based on the feedback data of the slave arm joint forces. When a task is to be performed in the autonomous mode, reference inputs to the slave arm control system can be generated by a *path planner*. The reference inputs can be expressed in joint space or in Cartesian space. Following the convention in [31], 8 coordinate frames are assigned to the manipulator as illustrated in Figure 3 showing the manipulator in its home configuration with all joint angles being zero. Each i th frame $\{i\}$ is characterized by its coordinate axes $\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i$ and its origin $\mathbf{0}_i$ for $i = 0, 1, 2, \dots, 7$. The Denavit-Hartenberg parameters for the assigned coordinate frames are listed in Table 1 given below:

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0°	0.000in	0.0in	θ_1
2	-90°	0.000in	0.0in	θ_2
3	90°	5.625in	27.0in	θ_3
4	-90°	4.250in	0.0in	θ_4
5	90°	-4.250in	27.0in	θ_5
6	-90°	3.125in	0.0in	θ_6
7	90°	-3.125in	0.0in	θ_7

Table 1: D-H parameters of the RRC K-1607 manipulator.

3 The Forward Kinematic Transformation

This section considers the forward kinematic transformation for the above slave arm, which can be used in a Cartesian-space control scheme to transform the 7 joint angles θ_i for $i=1, 2, \dots, 7$ of the slave arm into the corresponding position and orientation, referred here to as *configuration* of the manipulator end-effector frame, Frame $\{7\}$, with respect to the base frame, Frame $\{0\}$. The configuration of the i th frame with respect to the $(i-1)$ th frame is represented by the following homogeneous transformation matrix:

$${}^{i-1}\mathbf{T} = \begin{bmatrix} {}^{i-1}\mathbf{R} & {}^{i-1}\mathbf{p} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (1)$$

$$= \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_{i-1} \\ \sin \theta_i \cos \alpha_{i-1} & \cos \theta_i \cos \alpha_{i-1} & -\sin \alpha_{i-1} & -d_i \sin \alpha_{i-1} \\ \sin \theta_i \sin \alpha_{i-1} & \cos \theta_i \sin \alpha_{i-1} & \cos \alpha_{i-1} & d_i \cos \alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

for $i=1, 2, \dots, 7$ where ${}^{i-1}\mathbf{R}$ and ${}^{i-1}\mathbf{p}$ represent the orientation and position of the i th frame expressed in the $(i-1)$ th frame, respectively. The transformation ${}^0\mathbf{T}_7$ consisting of the orientation matrix ${}^0\mathbf{R}_7$ and the position vector ${}^0\mathbf{p}_7$ expresses the configuration of Frame $\{7\}$ with respect to Frame $\{0\}$ and is computed by

$${}^0\mathbf{T}_7 = {}^0\mathbf{T}_1 {}^1\mathbf{T}_2 {}^2\mathbf{T}_3 {}^3\mathbf{T}_4 {}^4\mathbf{T}_5 {}^5\mathbf{T}_6 {}^6\mathbf{T}_7. \quad (3)$$

Carrying out the matrix multiplications in (3) and performing intensive trigonometric simplifications we obtain

$${}^0_7\mathbf{T} = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

where

$$\left. \begin{aligned} n_x &= s_7 h_1 + c_7 j_1 \\ s_x &= c_7 h_1 - s_7 j_1 \\ a_x &= s_6 h_2 + c_6 g_2 \\ p_x &= a_6 j_1 + a_5 h_2 + d_5(s_4 f_1 + c_1 s_2 c_4) \\ &\quad + a_4 g_1 + a_3 f_2 + c_1 j_3 \end{aligned} \right\} \quad (5)$$

$$\left. \begin{aligned} n_y &= s_7 h_3 + c_7 j_2 \\ s_y &= c_7 h_3 - s_7 j_2 \\ a_y &= s_6 h_4 + c_6 g_4 \\ p_y &= a_6 j_2 + a_5 h_4 + d_5(s_4 f_4 + s_1 s_2 c_4) \\ &\quad + a_4 g_3 + a_3 f_4 + s_1 j_3 \end{aligned} \right\} \quad (6)$$

$$\left. \begin{aligned} n_z &= s_7 g_5 + c_7 h_5 \\ s_z &= c_7 g_5 - s_7 h_5 \\ a_z &= s_6 g_6 + c_6 f_6 \\ p_z &= a_6 h_5 + a_5 g_6 + d_5(c_2 c_4 - s_2 c_3 s_4) \\ &\quad + a_4 f_5 - a_3 s_2 c_3 - a_2 s_2 + d_3 c_2 \end{aligned} \right\} \quad (7)$$

$$\left. \begin{aligned} f_1 &= -c_1 c_2 s_3 - s_1 c_3 \\ f_2 &= c_1 c_2 c_3 - s_1 s_3 \\ f_3 &= -s_1 c_2 s_3 - c_1 c_3 \\ f_4 &= s_1 c_2 c_3 + c_1 s_3 \\ f_5 &= -s_2 c_3 c_4 - c_2 s_4 \\ f_6 &= -s_1 c_3 s_4 + c_2 c_4 \end{aligned} \right\} \quad (8)$$

$$\left. \begin{aligned} g_1 &= -c_1 s_2 s_4 + c_2 f_2 \\ g_2 &= c_1 s_2 c_4 + s_4 f_2 \\ g_3 &= -s_1 s_2 s_4 + c_4 f_4 \\ g_4 &= s_1 s_2 c_4 + s_4 f_4 \\ g_5 &= s_2 s_3 c_5 - s_5 f_5 \\ g_6 &= s_2 s_3 s_5 + c_5 f_5 \end{aligned} \right\} \quad (9)$$

$$\left. \begin{aligned} h_1 &= c_5 f_1 - s_5 g_1 \\ h_2 &= s_5 f_1 + c_5 g_1 \\ h_3 &= c_5 f_3 - s_5 g_3 \\ h_4 &= s_5 f_3 + c_5 g_3 \\ h_5 &= c_6 g_6 - s_6 f_6 \end{aligned} \right\} \quad (10)$$

$$\left. \begin{aligned} j_1 &= c_6 h_2 - s_6 g_2 \\ j_2 &= c_6 h_4 - s_6 g_4 \\ j_3 &= d_3 s_2 + a_2 c_2, \end{aligned} \right\} \quad (11)$$

and we have used the compact notations, $c_i \equiv \cos \theta_i$ and $s_i \equiv \sin \theta_i$. It is also noted that in (5)-(11), a_{i-1} and d_i for $i=1,2,\dots,7$ are manipulator parameters listed in Table 1. Since matrix multiplications are avoided in (5)-(11), the computation time required for the above forward kinematic transformation is greatly reduced. Consequently, the derived forward kinematic equations are highly suitable for real-time control implementation.

4 Differential Motion Analysis

This section is devoted to the analysis of the slave arm differential motion. In the following, we first compute the manipulator Jacobian using the vector cross product method and then discuss its inverse computation using the method of Moore-Penrose pseudo-inverse. After that, we review the quaternion representation of orientation which will be used in the computer simulation study.

4.1 The Manipulator Jacobian

To be compatible with the coordinate frame assignments according to the convention given in [31], the vector cross product method [11] is modified and applied to derive the manipulator Jacobian. According to [11] the manipulator Jacobian is obtained by

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 & \mathbf{J}_2 & \mathbf{J}_3 & \mathbf{J}_4 & \mathbf{J}_5 & \mathbf{J}_6 & \mathbf{J}_7 \end{bmatrix} \quad (12)$$

where

$$\mathbf{J}_i = \begin{bmatrix} \mathbf{b}_i \times \mathbf{p}_i \\ \mathbf{b}_i \end{bmatrix}, \quad i=1,2,\dots,7 \quad (13)$$

and \mathbf{b}_i , defined as the unit vector pointing along the axis of motion of Joint i expressed in Frame $\{0\}$, is given by

$$\mathbf{b}_i = {}^0\mathbf{R}_1 {}^1\mathbf{R}_2 \dots {}^{i-1}\mathbf{R}_i \mathbf{b}_0, \quad i=1,2,\dots,7 \quad (14)$$

with

$$\mathbf{b}_0 = [0 \ 0 \ 1]^T \quad (15)$$

and \mathbf{p}_i , defined as the vector pointing from the origin of the i th-frame to the origin of Frame $\{7\}$, expressed in Frame $\{0\}$, is obtained from

$$\begin{bmatrix} \mathbf{p}_i \\ 1 \end{bmatrix} = {}^0\mathbf{T}_7 \mathbf{x}_0 - {}^0\mathbf{T}_i \mathbf{x}_0, \quad i=1,2,\dots,7 \quad (16)$$

with

$$\mathbf{x}_0 = [0 \ 0 \ 0 \ 1]^T \quad (17)$$

and \times indicates the vector cross product. A Fortran program was written to compute the manipulator Jacobian \mathbf{J} whose first three columns are presented below:

$$\mathbf{J}_1 = \begin{bmatrix} -p_y \\ -p_x \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}; \quad \mathbf{J}_2 = \begin{bmatrix} c_1 p_z \\ s_1 p_z \\ -s_1 p_y - c_1 p_x \\ -s_1 \\ c_1 \\ 0 \end{bmatrix} \quad (18)$$

and

$$\mathbf{J}_3 = \begin{bmatrix} s_1 s_2 (p_z + a_2 s_2 - d_3 c_2) - c_2 (p_y - a_2 s_1 c_2 - d_3 s_1 s_2) \\ -c_1 s_2 (p_z + a_2 s_2 - d_3 c_2) + c_2 (p_x - a_2 c_1 c_2 - d_3 c_1 s_2) \\ c_1 s_2 (p_y - a_2 s_1 c_2 - d_3 s_1 s_2) - s_1 s_2 (p_x - a_2 c_1 c_2 - d_3 c_1 s_2) \\ c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix}. \quad (19)$$

4.2 The Jacobian Inverse

The Cartesian velocity vector $\dot{\mathbf{x}}(t)$ are related to joint angle velocity vector $\dot{\mathbf{q}}(t)$ by the Jacobian \mathbf{J} as

$$\dot{\mathbf{x}}(t) = \mathbf{J}\dot{\mathbf{q}}(t). \quad (20)$$

The inverse solution to (20) which minimizes the weighted quadratic form $\dot{\mathbf{q}}^T \mathbf{W}^{-1} \dot{\mathbf{q}}$, is given by [14]

$$\dot{\mathbf{q}}(t) = \mathbf{J}_W^\dagger \dot{\mathbf{x}}(t) + (\mathbf{I}_7 - \mathbf{J}_W^\dagger \mathbf{J}) \mathbf{z} \quad (21)$$

where \mathbf{J}_W^\dagger , the *Weighted Pseudo-Inverse* of the Jacobian \mathbf{J} is given by

$$\mathbf{J}_W^\dagger = \mathbf{W} \mathbf{J}^T [\mathbf{J} \mathbf{W} \mathbf{J}^T]^{-1} \quad (22)$$

\mathbf{W} , the *Weighting Matrix* is a symmetric matrix, \mathbf{z} denotes an arbitrary joint velocity vector and the second term of (21) belongs to the null space of \mathbf{J} . Vector \mathbf{z} can be selected for optimization purposes. When $\mathbf{W} = \mathbf{I}$ and $\mathbf{z} = 0$, then (22) reduces to the well-known *Moore-Penrose Pseudo-Inverse* of the Jacobian given by

$$\mathbf{J}^\dagger = \mathbf{J}^T (\mathbf{J} \mathbf{J}^T)^{-1} \quad (23)$$

which provides the minimum norm least-squares solution.

4.3 Quaternion Representation

Quaternion has gained more popularity than Roll-Pitch-Yaw Angles in representing manipulator orientation because Roll-Pitch-Yaw Angles suffer from singularities and computational complexities [13]. The *Quaternion* consisting of a scalar η and a vector $\mathbf{s} =$

$[\beta \ \gamma \ \xi]^T$, also called *Euler Parameters* of an orientation matrix \mathbf{R} specified by

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (24)$$

is obtained by an operator \mathbf{Q} defined by

$$(\eta, \mathbf{s}) = \mathbf{Q}\{\mathbf{R}\} \quad (25)$$

such that

$$\begin{aligned} \eta &= \sqrt{1 + r_{11} + r_{22} + r_{33}}/2 \\ \beta &= (r_{32} - r_{23})/4\eta \\ \gamma &= (r_{13} - r_{31})/4\eta \\ \xi &= (r_{21} - r_{12})/4\eta. \end{aligned} \quad (26)$$

On the other hand, an orientation matrix \mathbf{R} can be computed from its quaternion by the inverse operator defined by

$$\mathbf{R} = \mathbf{Q}^{-1}\{\eta, \mathbf{s}\} \quad (27)$$

so that

$$\mathbf{R} = (\eta^2 - \mathbf{s}^T \mathbf{s}) \mathbf{I}_6 + 2\mathbf{s}\mathbf{s}^T - 2\eta \mathbf{s}^\times \quad (28)$$

where

$$\mathbf{s}^\times = \begin{bmatrix} 0 & -\xi & \gamma \\ \xi & 0 & -\beta \\ -\gamma & \beta & 0 \end{bmatrix}. \quad (29)$$

Now considering two orientation matrices ${}^0_1\mathbf{R}$ and ${}^0_2\mathbf{R}$ which represent the orientation of Frame $\{1\}$ and $\{2\}$ with respect to Frame $\{0\}$, respectively, we can write

$${}^0_2\mathbf{R} = {}^0_1\mathbf{R} \ {}^1_2\mathbf{R}. \quad (30)$$

In (30), since ${}^1_2\mathbf{R}$ is postmultiplied to ${}^0_1\mathbf{R}$, ${}^1_2\mathbf{R}$ represents a rotation of Frame $\{1\}$ about Frame $\{1\}$ to move Frame $\{1\}$ to Frame $\{2\}$. ${}^1_2\mathbf{R}$ can also be interpreted as the orientation of Frame $\{2\}$ with respect to Frame $\{1\}$. However if the rotation is performed about Frame $\{0\}$, then we should write

$${}^0_2\mathbf{R} = \overline{{}^1_2\mathbf{R}} \ {}^0_1\mathbf{R} \quad (31)$$

where $\overline{{}^1_2\mathbf{R}}$ represents the rotation of Frame $\{1\}$ about Frame $\{0\}$ to bring Frame $\{1\}$ to Frame $\{2\}$ and can be computed from (31) as

$$\overline{{}^1_2\mathbf{R}} = {}^0_2\mathbf{R} \ {}^0_1\mathbf{R}^{-1} = {}^0_2\mathbf{R} \ {}^0_1\mathbf{R}^T. \quad (32)$$

Suppose (η_1, \mathbf{s}_1) and (η_2, \mathbf{s}_2) are the quaternions of ${}^0_1\mathbf{R}$ and ${}^0_2\mathbf{R}$, respectively. Then the quaternion of ${}^1_2\mathbf{R}$ can be expressed in terms of those of ${}^0_1\mathbf{R}$ and ${}^0_2\mathbf{R}$ as follows:

$$\delta\eta = \eta_1\eta_2 + \mathbf{s}_1^T \mathbf{s}_2 \quad (33)$$

and

$$\delta \mathbf{s} = \eta_1 \mathbf{s}_2 - \eta_2 \mathbf{s}_1 + \mathbf{s}_1^\times \mathbf{s}_2. \quad (34)$$

Now we are interested in finding how the quaternion of $\overline{{}_2^1\mathbf{R}}$ are related to differential rotations introduced in [15]. According to [15], if the orientation difference between Frame {1} and Frame {2} is small then

$${}_2^0\mathbf{R} \approx \begin{bmatrix} 1 & -\delta_z & -\delta_y \\ \delta_z & 1 & -\delta_x \\ -\delta_y & \delta_x & 1 \end{bmatrix} {}_1^0\mathbf{R} \quad (35)$$

where δ_x , δ_y , and δ_z denote the differential rotations of Frame {1} made in any order about the x, y, and z axes of Frame {0}, respectively to bring Frame {1} to Frame {2}.

A comparison of (31) and (35) yields

$$\overline{{}_2^1\mathbf{R}} \approx \begin{bmatrix} 1 & -\delta_z & -\delta_y \\ \delta_z & 1 & -\delta_x \\ -\delta_y & \delta_x & 1 \end{bmatrix} \quad (36)$$

from which differential rotations δ_x , δ_y , and δ_z can be computed from the quaternion of $\overline{{}_2^1\mathbf{R}}$ by taking the quaternion on both sides of (36) using (26) and solving for δ_x , δ_y , and δ_z as follows:

$$\begin{aligned} \delta_x &\approx 2\delta\beta \\ \delta_y &\approx 2\delta\gamma \\ \delta_z &\approx 2\delta\xi \end{aligned} \quad (37)$$

Equation (37) can be employed to compute the rotation velocities with a relatively good accuracy provided that the quaternion of $\overline{{}_2^1\mathbf{R}}$ is given.

5 Computer Simulation Study

This section presents the results of the computer simulation study conducted to verify the above mathematical developments. The study is composed mainly of three parts, the first part is devoted to investigate the efficiency of the derived Jacobian in converting joint angle velocities to Cartesian velocities, the second to evaluate the accuracy of the pseudo-inverse Jacobian and the third to verify the equivalence between Cartesian velocities and quaternion representation. Computer simulation is repeated for various sampling times so that a maximum permitted sampling time can be established for an acceptable conversion accuracy. English units will be used to present the results.

5.1 Part 1: Joint to Cartesian Velocities

Figure 4 illustrates the computer simulation scheme used for Part 1 and Part 2. In the upper loop, a set of test joint angle trajectories are converted to the corresponding Cartesian

trajectories using the derived forward kinematic transformation. The orientation matrix ${}^0_7\mathbf{R}$ is used to compute the differential rotations by employing (35). In the lower loop, the joint velocities which are obtained by differentiating the test joint angle trajectories are supplied to the Jacobian which produces the corresponding Cartesian velocities. The Cartesian velocities obtained from the upper loop are then compared with those from the lower loop to compute the conversion errors. Figure 5 shows the error between the x-axis velocities \dot{p}_{xJ} (from Jacobian) and \dot{p}_x for two different sampling times. The maximum error is about 0.5 inch/sec for a sampling time of 10 msec (indicated by solid line) and about 6 inch/sec for a sampling time of 100 msec (indicated by asterisk line). Figure 5 presents the error between the x-axis angular velocities ω_{xJ} and ω_x for sampling times of 10 msec (solid line) and 100 msec (dotted line). The maximum angular velocity errors are about 0.5 miliinch/sec and 5 miliinch/sec for sampling times of 10 msec and 100 msec, respectively.

5.2 Part 2: Cartesian to Joint Velocities

In the lower loop of Figure 4, the Cartesian velocities provided by the Jacobian are supplied to the Jacobian pseudo-inverse which is computed by Equation (23) and whose outputs are compared with the joint velocities. Figures 7 and 8 show the joint angle velocities $\dot{\theta}_{1J}$ (from the pseudo-inverse) and $\dot{\theta}_1$ for sampling times of 10 msec and 100 msec, respectively. According to the obtained results, the pseudo-inverse does not provide adequate conversion of Cartesian velocities to joint velocities at a sampling time of 100 msec. The velocity conversion is excellent at a sampling time of 10 msec.

5.3 Part 3: Quaternion Representation

Figure 9 illustrates the computer simulation scheme used to verify the equivalence between Cartesian velocities and quaternion representation. In the upper loop of Figure 9, using Equations (33)-(34), we compute the quaternion of the orientation difference given by

$$\Delta {}^0_7\mathbf{R} = {}^0_7\mathbf{R}(t_i) {}^0_7\mathbf{R}^T(t_{i-1}) \quad (38)$$

where ${}^0_7\mathbf{R}(t_i)$ denotes the orientation matrix evaluated at the i th sampling during the computer simulation. In the lower loop, the quaternion can be computed from the output of the Jacobian by employing Equation (37) and then compared with the quaternion of the upper loop to determine the deviations. Figures 10 and 11 show the simulation results of the errors of $\delta\beta$ (solid line) and $\delta\gamma$ (asterisk line) for sampling times of 10 msec and 100 msec, respectively. In the case of 100 msec sampling time, the maximum errors for $\delta\beta$ and $\delta\gamma$ are 15 miliinch/sec and 0.15 miliinch/sec, respectively and are negligible in the case of the sampling time of 10ms.

6 Proposed Control Schemes

This section considers the problem of controlling the compliant and non-compliant motion of the slave arm end-effector. When the slave arm performs non-compliant motion, i.e. without being in contact with the environment, it is sufficient to employ *pure position control schemes* whose error-correcting forces are computed based only on the position errors. However during a compliant motion mode in which the slave arm end-effector is constantly in contact with the environment a *hybrid position/force² control scheme* which controls not only the position of the end-effector but also the contact forces it applies on the environment, should be applied. In the following, we present and discuss three control schemes which have been under study for controlling the slave arm motion and briefly report some preliminary findings.

6.1 Joint-Space Adaptive Control Scheme

Figure 12 shows the organization of a joint-space control scheme which has been considered for controlling the non-compliant motion of the slave arm. In the control scheme, actual joint angles measured by 7 joint sensors are compared with desired joint angles which are obtained from desired configuration of the slave arm end-effector through the inverse kinematics. The joint variable errors then serve as inputs to a set of proportional-derivative (**PD**)- controllers whose gains are adjusted by an adaptation law so that the error-correcting joint forces provided by the controllers track the slave arm end-effector along a desired path. The adaptation law was derived using the Lyapunov theory and the concept of model reference adaptive control (**MRAC**) under the assumption that the slave arm performs *slowly varying* motion. From the fact that the derived adaptation law does not have to evaluate the slave arm dynamics, it is computationally fast and very attractive to real-time control. Computer simulation results reported in [6] showed that the slave arm end-effector under the control of the above scheme can track several test paths with minimal tracking errors under sudden change in payload. The developed joint-space control scheme is currently implemented by GSFC for real-time control of the slave arm motion.

6.2 Cartesian-Space Adaptive Control Scheme

An adaptive control scheme in Cartesian space is presented in Figure 13. As the figure shows, feedback information of the actual joint variables are converted into the corresponding Cartesian variables by the forward kinematic transformation. The actual Cartesian variables are then compared with the desired Cartesian variables representing the desired configuration of the slave arm end-effector, and the corresponding Cartesian errors are supplied to a set of PD-controllers whose gains are adjusted by an adaptation law. The adaptation law is designed such that the joint forces which are obtained by transforming

²In this report, “position” implies both “position and orientation” and “force” both “force and torque”.

the Cartesian forces produced by the adaptive PD controllers using the Jacobian transpose will track the end-effector along desired paths. Extending the development in [6], an adaptation law was derived and presented in [5] under the assumption of slowly-varying motion. Computer simulation study is conducted to investigate the performance of the Cartesian-space control scheme and simulation results reported in [7] showed that the Cartesian adaptive controller can provide excellent tracking of several test paths with minimal tracking errors under both constant and time-varying payloads.

6.3 Hybrid Position/Force Control Scheme

Figure 14 presents a hybrid position/force control scheme whose structure is similar to that introduced in [12] except that the controller gains of the current control scheme are adjusted by an adaptation law. As Figure 14 shows, the control scheme mainly consists of two control loops, the upper loop for position and the lower for force control. A (6x6) diagonal *compliance selection matrix* S whose main diagonal elements s_{ii} for $i = 1, 2, \dots, 6$ assume either 1 or 0, allows the user to select which DOF to be position-controlled and which to be force-controlled by setting the element s_{ii} properly, namely $s_{ii} = 1$ for the i th DOF to be force-controlled and $s_{ii} = 0$ for the i th DOF to be position-controlled. In other words, the hybrid position/force control scheme allows independent and simultaneous control of position and force. The adaptation law which adjusts the gains of the PD-controllers of the position and force control loops so that the end-effector can follow a desired path while applying desired contact forces on the environment despite disturbances such as varying environment stiffness, was developed in [8]. Simulation results showed that the control scheme provided remarkable performance in simultaneous position/force control for both constant and variable stiffness cases.

7 The Cartesian Adaptive Scheme

This section is devoted to present the development of the Cartesian-space adaptive control scheme proposed in Section 6.2 and illustrated in Figure 13. First using the MRAC and Lyapunov theory, the adaptation scheme will be developed. Computer simulation will then be conducted to evaluate the performance of the developed adaptation scheme in tracking several test paths. Matrix and vector notations used in this section are listed below:

- \mathbf{M}^T : transpose of the matrix \mathbf{M}
- $\mathbf{0}_n$: (nxn) matrix whose elements are all zero
- \mathbf{I}_n : (nxn) identity matrix
- $tr[\mathbf{M}]$: trace of matrix \mathbf{M} .

7.1 Adaptation Scheme Development

The dynamics of a 7 DOF redundant manipulator performing non-compliant motion can be expressed in Cartesian space as [23]

$$\mathbf{F}(t) = \mathbf{A}(\mathbf{x}, \dot{\mathbf{x}}) \ddot{\mathbf{x}}(t) + \mathbf{\Phi}(\mathbf{x}, \dot{\mathbf{x}}) \dot{\mathbf{x}}(t) + \mathbf{\Gamma}(\mathbf{x}, \dot{\mathbf{x}}) \mathbf{x}(t) \quad (39)$$

where $\mathbf{x}(t)$, $\dot{\mathbf{x}}(t)$ and $\ddot{\mathbf{x}}(t)$ denote the (6x1) vectors of the manipulator end-effector Cartesian position, velocity and acceleration, respectively and $\mathbf{F}(t)$, the (6x1) Cartesian force applied to the end-effector. $\mathbf{A}(\mathbf{x}, \dot{\mathbf{x}})$, a (6x6) symmetric positive-definite matrix, is the Cartesian mass matrix, $\mathbf{\Phi}(\mathbf{x}, \dot{\mathbf{x}})$ and $\mathbf{\Gamma}(\mathbf{x}, \dot{\mathbf{x}})$ are (6x6) matrices whose elements are highly complex nonlinear functions of \mathbf{x} and $\dot{\mathbf{x}}$. $\mathbf{\Phi}(\mathbf{x}, \dot{\mathbf{x}}) \dot{\mathbf{x}}(t)$ and $\mathbf{\Gamma}(\mathbf{x}, \dot{\mathbf{x}}) \mathbf{x}(t)$ represent the Cartesian Coriolis and centrifugal force vector and the Cartesian gravity loading vector, respectively.

Consider now a PD controller with time-varying gains, defined by

$$\mathbf{F}(t) = \mathbf{K}_p(t) \mathbf{x}_e(t) + \mathbf{K}_d(t) \dot{\mathbf{x}}_e(t) \quad (40)$$

where $\mathbf{x}_e(t)$ given by

$$\mathbf{x}_e(t) = \mathbf{x}_d(t) - \mathbf{x}(t) \quad (41)$$

denotes the Cartesian error vector between the actual Cartesian position vector $\mathbf{x}(t)$ and the desired Cartesian position vector $\mathbf{x}_d(t)$, $\mathbf{K}_p(t)$ and $\mathbf{K}_d(t)$ are gain matrices of the proportional and derivative controllers, respectively.

Substituting (40) into (39) yields

$$\mathbf{A} \ddot{\mathbf{x}}_e + (\mathbf{\Phi} + \mathbf{K}_d) \dot{\mathbf{x}}_e + (\mathbf{\Gamma} + \mathbf{K}_p) \mathbf{x}_e = \mathbf{A} \ddot{\mathbf{x}}_d + \mathbf{\Phi} \dot{\mathbf{x}}_d + \mathbf{\Gamma} \mathbf{x}_d \quad (42)$$

where the dependent variables of the matrices and vectors were dropped for simplicity.

The state-space representation of (42) can be obtained by defining a (12x1) state variable vector $\mathbf{z}(t)$

$$\mathbf{z}(t) = [\mathbf{x}_e^T(t) \quad \dot{\mathbf{x}}_e^T(t)]^T \quad (43)$$

so that (42) can be converted to

$$\dot{\mathbf{z}}(t) = \begin{bmatrix} \mathbf{0}_6 & \mathbf{0}_6 \\ -\mathbf{B}_1 & -\mathbf{B}_2 \end{bmatrix} \mathbf{z}(t) + \begin{bmatrix} \mathbf{0}_6 & \mathbf{0}_6 & \mathbf{0}_6 \\ \mathbf{B}_3 & \mathbf{B}_4 & \mathbf{I}_6 \end{bmatrix} \mathbf{u}(t) \quad (44)$$

where

$$\mathbf{B}_1 = \mathbf{A}^{-1}(\mathbf{\Gamma} + \mathbf{K}_p), \quad \mathbf{B}_2 = \mathbf{A}^{-1}(\mathbf{\Phi} + \mathbf{K}_d), \quad (45)$$

and

$$\mathbf{B}_3 = \mathbf{A}^{-1} \mathbf{\Gamma}, \quad \mathbf{B}_4 = \mathbf{A}^{-1} \mathbf{\Phi}, \quad (46)$$

and

$$\mathbf{u}(t) = [\mathbf{x}_d^T(t) \quad \dot{\mathbf{x}}_d^T(t) \quad \ddot{\mathbf{x}}_d^T(t)]^T. \quad (47)$$

In the framework of model reference adaptive control [30], (44) represents an *adjustable system*. In order to minimize the computational burden on real-time control, a *reference model* which characterizes the desired manipulator performance, should be selected as

$$\ddot{x}_{ei}(t) + 2 \xi_i \omega_i \dot{x}_{ei}(t) + \omega_i^2 x_{ei}(t) = 0 \text{ for } i=1,2,\dots,7 \quad (48)$$

where ξ_i and ω_i denote the damping ratio and the natural frequency of x_{ei} , and $x_{ei}(t)$ for $i=1,2,\dots,7$ are the elements of the tracking error vector $\mathbf{x}_e(t)$ defined by

$$\mathbf{x}_e(t) = [x_{e1}(t) \ x_{e2}(t) \ \dots \ x_{e6}(t)]^T. \quad (49)$$

As seen from Equation (48), the reference model is a linear time-invariant system consisting of 6 uncoupled systems and the i th system represents the desired behavior of error in the i th Cartesian position.

From (48), the state presentation of the reference model can be obtained as

$$\dot{\mathbf{z}}_m(t) = \mathbf{\Delta} \mathbf{z}_m(t) = \begin{bmatrix} \mathbf{0}_6 & \mathbf{I}_6 \\ -\mathbf{\Delta}_1 & -\mathbf{\Delta}_2 \end{bmatrix} \mathbf{z}_m(t), \quad (50)$$

where $\mathbf{\Delta}_1 = \text{diag}(\omega_i^2)$ and $\mathbf{\Delta}_2 = \text{diag}(2\xi_i\omega_i)$ are constant (6x6) diagonal matrices, and

$$\mathbf{z}_m(t) = [\mathbf{x}_m^T(t) \ \dot{\mathbf{x}}_m^T(t)]^T \quad (51)$$

with

$$\mathbf{x}_m = (x_{e1} \ x_{e2} \ \dots \ x_{e6})^T. \quad (52)$$

Now solving (50), we obtain

$$\mathbf{z}_m(t) = e^{\mathbf{\Delta}t} \mathbf{z}_m(0) \quad (53)$$

where the initial value of $\mathbf{z}_m(t)$ is denoted by $\mathbf{z}_m(0)$. Here if we assume that the initial values of the actual and desired Cartesian position and velocity vectors are identical, i.e. $\mathbf{z}_m(0) = 0$, then $\mathbf{z}_m(t) = 0$. Otherwise we can properly select ξ_i and ω_i so that all eigenvalues of $\mathbf{\Delta}$ are stable to make $\mathbf{z}_m(t) \rightarrow 0$ as $t \rightarrow \infty$.

Next if an adaptation error vector $\mathbf{E}(t)$ is defined as

$$\mathbf{E}(t) = \mathbf{z}_m(t) - \mathbf{z}(t), \quad (54)$$

then from (44) and (50), we can obtain an *error system* whose dynamics is represented by

$$\begin{aligned} \dot{\mathbf{E}}(t) = & \begin{bmatrix} \mathbf{0}_6 & \mathbf{I}_6 \\ -\mathbf{\Delta}_1 & -\mathbf{\Delta}_2 \end{bmatrix} \mathbf{E}(t) + \begin{bmatrix} \mathbf{0}_6 & \mathbf{0}_6 \\ \mathbf{B}_1 - \mathbf{\Delta}_1 & \mathbf{B}_2 - \mathbf{\Delta}_2 \end{bmatrix} \mathbf{z}(t) \\ & + \begin{bmatrix} \mathbf{0}_6 & \mathbf{0}_6 & \mathbf{0}_6 \\ -\mathbf{B}_3 & -\mathbf{B}_4 & -\mathbf{I}_6 \end{bmatrix} \mathbf{u}(t). \end{aligned} \quad (55)$$

We proceed to select a Lyapunov function $v(t)$ given by

$$v(t) = \mathbf{E}^T \mathbf{P} \mathbf{E} + tr \left[(\mathbf{B}_1 - \mathbf{\Delta}_1)^T \mathbf{\Psi}_1 (\mathbf{B}_1 - \mathbf{\Delta}_1) \right] \\ + tr \left[(\mathbf{B}_2 - \mathbf{\Delta}_2)^T \mathbf{\Psi}_2 (\mathbf{B}_2 - \mathbf{\Delta}_2) \right] + tr [\mathbf{B}_3^T \mathbf{\Psi}_3 \mathbf{B}_3] + tr [\mathbf{B}_4^T \mathbf{\Psi}_4 \mathbf{B}_4], \quad (56)$$

where \mathbf{P} and $\mathbf{\Psi}_i$ for $i=1,2,\dots,4$, are positive definite matrices which will be determined.

Differentiating (56) with respect to time and extensively simplifying the resulting expression yield

$$\dot{v}(t) = \mathbf{E}^T (\mathbf{P} \mathbf{\Delta} + \mathbf{\Delta}^T \mathbf{P}) \mathbf{E} + 2tr \left[(\mathbf{B}_1 - \mathbf{\Delta}_1)^T (\mathbf{\Upsilon} \mathbf{x}_e^T + \mathbf{\Psi}_1 \dot{\mathbf{B}}_1) \right] - 2tr \left[\mathbf{B}_3^T (\mathbf{\Upsilon} \mathbf{x}_d^T - \mathbf{\Psi}_3 \dot{\mathbf{B}}_3) \right] \\ + 2tr \left[(\mathbf{B}_2 - \mathbf{\Delta}_2)^T (\mathbf{\Upsilon} \mathbf{x}_e^T + \mathbf{\Psi}_2 \dot{\mathbf{B}}_2) \right] - 2tr \left[\mathbf{B}_4^T (\mathbf{\Upsilon} \mathbf{x}_d^T - \mathbf{\Psi}_4 \dot{\mathbf{B}}_4) \right] \quad (57)$$

where

$$\mathbf{\Upsilon} = [\mathbf{P}_2 \ \mathbf{P}_3] \mathbf{z}(t) = -\mathbf{P}_2 \mathbf{x}_e - \mathbf{P}_3 \dot{\mathbf{x}}_e \quad (58)$$

and \mathbf{P} is given by

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_2 \\ \mathbf{P}_2 & \mathbf{P}_3 \end{bmatrix} \quad (59)$$

and it is noted that $\mathbf{E}(t) = -\mathbf{z}(t)$ since $\mathbf{z}_m(t) = 0$ based on our previous assumption.

Now if ξ_i and ω_i of (48) are selected so that $\mathbf{\Delta}$ is a *Hurwitz* matrix [30], i.e. all eigenvalues of $\mathbf{\Delta}$ have negative real parts, then according to *Lyapunov Theorem*, there exists a positive definite symmetric matrix \mathbf{P} satisfying the Lyapunov equation given by

$$\mathbf{P} \mathbf{\Delta} + \mathbf{\Delta}^T \mathbf{P} = -\mathbf{Q} \quad (60)$$

for any given positive-definite symmetric matrix \mathbf{Q} .

Now in (57) letting

$$\mathbf{\Upsilon} \mathbf{x}_e^T + \mathbf{\Psi}_1 \dot{\mathbf{B}}_1 = \mathbf{\Upsilon} \dot{\mathbf{x}}_e^T + \mathbf{\Psi}_2 \dot{\mathbf{B}}_2 = 0, \quad (61)$$

$$\mathbf{\Upsilon} \mathbf{x}_d^T - \mathbf{\Psi}_3 \dot{\mathbf{B}}_3 = \mathbf{\Upsilon} \dot{\mathbf{x}}_d^T - \mathbf{\Psi}_4 \dot{\mathbf{B}}_4 = 0, \quad (62)$$

makes (57) become

$$\dot{v}(t) = -\mathbf{E}^T \mathbf{Q} \mathbf{E} \quad (63)$$

which is a negative definite function of $\mathbf{E}(t)$. Also from (61)-(62), we obtain

$$\dot{\mathbf{B}}_1 = -\mathbf{\Psi}_1^{-1} \mathbf{\Upsilon} \mathbf{x}_e^T; \quad \dot{\mathbf{B}}_2 = -\mathbf{\Psi}_2^{-1} \mathbf{\Upsilon} \dot{\mathbf{x}}_e^T, \quad (64)$$

$$\dot{\mathbf{B}}_3 = \mathbf{\Psi}_3^{-1} \mathbf{\Upsilon} \mathbf{x}_d^T; \quad \dot{\mathbf{B}}_4 = \mathbf{\Psi}_4^{-1} \mathbf{\Upsilon} \dot{\mathbf{x}}_d^T. \quad (65)$$

In (60), \mathbf{P} can be made to be a positive definite matrix by properly selecting $\mathbf{\Delta}$ and \mathbf{Q} . Therefore, according to (56) the error system described in (55) is asymptotically stable, e.g. $\mathbf{E}(t)$ approaches zero asymptotically as $t \rightarrow \infty$ if we can show that $\mathbf{\Psi}_i$ for $i=1,2,\dots,4$, are also positive definite matrices. In other words, the *adjustable system* (44) will follow the reference model very closely, or $\mathbf{z}(t)$ approaches \mathbf{z}_m asymptotically as $t \rightarrow \infty$.

Now if we assume that the end-effector motion is slow compared to the sampling rate of updating the values of $\mathbf{K}_p(t)$ and $\mathbf{K}_d(t)$, then the manipulator dynamics can be considered *invariant* during the sampling interval of the controllers. In this case, the elements of \mathbf{A} , $\mathbf{\Phi}$ and $\mathbf{\Gamma}$ can be considered *nearly constant* during the sampling interval.

As a result, from (45) and (46) we get

$$\dot{\mathbf{B}}_1 \simeq \mathbf{A}^{-1} \dot{\mathbf{K}}_p, \quad (66)$$

$$\dot{\mathbf{B}}_2 \simeq \mathbf{A}^{-1} \dot{\mathbf{K}}_d, \quad (67)$$

$$\dot{\mathbf{B}}_3 \simeq 0; \quad \dot{\mathbf{B}}_4 \simeq 0. \quad (68)$$

Substituting (66)-(68) into (64)-(65) yields

$$\mathbf{A}^{-1} \dot{\mathbf{K}}_p = -\mathbf{\Psi}_1^{-1} \mathbf{\Upsilon} \mathbf{x}_e^T, \quad (69)$$

$$\mathbf{A}^{-1} \dot{\mathbf{K}}_d = -\mathbf{\Psi}_2^{-1} \mathbf{\Upsilon} \dot{\mathbf{x}}_e^T, \quad (70)$$

and

$$0 \simeq \mathbf{\Psi}_3^{-1} \mathbf{\Upsilon} \mathbf{x}_d^T; \quad 0 \simeq \mathbf{\Psi}_4^{-1} \mathbf{\Upsilon} \dot{\mathbf{x}}_d^T. \quad (71)$$

Now in (69)-(70), letting

$$\mathbf{\Psi}_1 = -\frac{1}{\beta_1} \mathbf{A}, \quad (72)$$

$$\mathbf{\Psi}_2 = -\frac{1}{\beta_2} \mathbf{A}, \quad (73)$$

where β_1 and β_2 are arbitrary positive scalars, and solving for $\dot{\mathbf{K}}_p$ and $\dot{\mathbf{K}}_d$, we arrive at

$$\dot{\mathbf{K}}_p = \beta_1 \mathbf{\Upsilon} \mathbf{x}_e^T, \quad (74)$$

$$\dot{\mathbf{K}}_d = \beta_2 \mathbf{\Upsilon} \dot{\mathbf{x}}_e^T. \quad (75)$$

In (72)-(73), obviously $\mathbf{\Psi}_1$ and $\mathbf{\Psi}_2$ are positive definite matrices that can be considered as nearly constant because \mathbf{A} is positive definite and slowly time-varying. In addition, $\mathbf{\Psi}_3$ and $\mathbf{\Psi}_4$ should be chosen to be positive definite matrices whose determinants approach ∞ in order to satisfy (71). To achieve this, we can select $\mathbf{\Psi}_3$ and $\mathbf{\Psi}_4$ to be diagonal matrices whose main diagonal elements assume very large and positive values.

Now integrating both sides of the equations given in (74)-(75) results in

$$\mathbf{K}_p(t) = \mathbf{K}_p(0) + \beta_1 \int_0^t (\mathbf{P}_2 \mathbf{x}_e + \mathbf{P}_3 \dot{\mathbf{x}}_e) \mathbf{x}_e^T dt \quad (76)$$

and

$$\mathbf{K}_d(t) = \mathbf{K}_d(0) + \beta_2 \int_0^t (\mathbf{P}_2 \mathbf{x}_e + \mathbf{P}_3 \dot{\mathbf{x}}_e) \dot{\mathbf{x}}_e^T dt \quad (77)$$

where $\mathbf{K}_p(0)$ and $\mathbf{K}_d(0)$ are initial conditions of $\mathbf{K}_p(t)$ and $\mathbf{K}_d(t)$, respectively and can be set arbitrarily.

The development of the adaptive controller is now completed. As shown in (76) and (77), the adaptation law is designed based on the errors of the Cartesian variables of the slave arm end-effector and the submatrices of \mathbf{P} . Therefore, the adaptation law is very computationally efficient because \mathbf{P} is a constant matrix and \mathbf{x}_e can be easily computed from the desired and actual Cartesian variables which are specified by the user and given from position sensors, respectively. The earlier assumption of slow end-effector motion compared to controller sampling rate is justified because high sampling rates up to 1KHz can be utilized in the implementation of the control scheme while the manipulator dynamics can be considered constant during each sampling interval of typically about 1 ms. Another attractive feature is that the implementation of the control scheme does not require the computation of the manipulator dynamics, which is very difficult, if not possible to model accurately.

7.2 Computer Simulation Study

This section is devoted to report results obtained from the computer simulation conducted to study the performance of the developed Cartesian-space adaptive control scheme which was applied to control the motion of a RRC K-1607 as shown in Figure 3. For the simulation, ξ_i and ω_i for $i=1,2$ are selected so that 2 characteristic roots of (11) are -1 and -2. Thus we have $\mathbf{D}_1=2\mathbf{I}_7$ and $\mathbf{D}_2=3\mathbf{I}_7$. Selecting $\mathbf{Q}_i = \mathbf{I}_{14}$ for $i=1,2,\dots,7$ and solving (22) gives $\mathbf{P}_2 = \mathbf{P}_3 = 0.25\mathbf{I}_7$. The adaptive controller gains are computed by substituting the derived values of \mathbf{P}_2 and \mathbf{P}_3 into (35)-(36) where $\mathbf{K}_p(0)$ and $\mathbf{K}_d(0)$ can be arbitrarily set. The scalars β_1 and β_2 can be adjusted to improve the tracking performance of the control scheme provided that their values are positive. In order to evaluate how the adaptive control scheme reacts to time-varying payload during the tracking of a desired path, the computer simulation is performed under sudden change in payload which is modeled by delayed step functions. The payload assumes zero at the beginning of the simulation, suddenly jumps to full payload of 10 lb at 1/3 of the simulation time and suddenly drops to zero again at 2/3 of the simulation time. Two study cases are considered: tracking a straight line and tracking a circular path during step changes in payload. A modified version of Manipulator Simulation Program (**MSP**) [32] is employed to simulate the dynamics of the RRC K-1607 manipulator. Fortran programs are written to implement the adaptive control laws to be used as a subroutine linked to the MSP program. The simulation is conducted on the DEC-VAX/80830 computer of GSFC with a sampling period of 10 msec and the simulation data are then imported into MATLAB for graphical presentation.

Study Case 1: Tracking A Straight Line

Computer simulation results of the case in which the robot end-effector is required to track a desired straight line in the x-y plane of the base frame from an initial position to a final position with desired velocity profiles, are presented in Figures 15-18. Such a

desired straight line path can be modeled by

$$x(t) = x_0 + 9[1 + 3e^{-\frac{3.5}{3}t} - 4e^{-\frac{3.5}{4}t}] \quad (78)$$

and

$$y(t) = y_0 + 9[2 + 6e^{-\frac{3.5}{3}t} - 8e^{-\frac{3.5}{4}t}] \quad (79)$$

where the initial position is specified by $x_0 = 33.996in$, $y_0 = 0in$ and the final position by $x_f = 42.996in$, $y_f = 18in$. According to Figures 15 and 16, the maximum value of errors in $x(t)$ and $y(t)$ are 0.3473in and 0.1599in, respectively. In addition, the root-mean-square (**RMS**) errors of $x(t)$ and $y(t)$ are 0.1536in and 0.1047in, respectively. Position errors in $z(t)$ and orientation errors are negligible according to the simulation results. As shown in Figure 17, the maximum velocity errors in $x(t)$ and $y(t)$ are -4.3450in/sec and -2.9749in/sec. The RMS errors of velocities in $x(t)$ and $y(t)$ are 0.2646 in/sec and 0.2282 in/sec, respectively. In Figure 18 the actual path the robot end-effector tracks is shown versus the desired one, and despite the abrupt change in payload, the tracking quality is quite remarkable.

Study Case 2: Tracking A Circular Path

Simulation results of the case in which the manipulator end-effector is required to track a desired circular path in the x-y plane of the base frame are reported in Figures 19-22. The circular path consisting of 3 segments is modeled by

$$x(t) = R \cos \Phi_i; \quad y(t) = R \sin \Phi_i \quad \text{for } t_{i-1} \leq t \leq t_i \quad (80)$$

for $i=1,2,3$ where the circular path radius $R = 24.32in$, and

$$\Phi_1(t) = \phi_0 + \frac{\beta}{2}t^2, \quad (81)$$

$$\Phi_2(t) = \phi_1 + \omega(t - t_1), \quad (82)$$

$$\Phi_3(t) = \phi_0 - \frac{\beta}{2}(t_3 - t)^2 \quad (83)$$

with $\phi_0 = 0in$; $\phi_1 = \Phi_1(t_1)$, angular velocity $\omega = \frac{2\pi}{9}radian/sec$ and the angular acceleration $\beta = \frac{2\pi}{9}radian/sec^2$. Figure 19 and 20 show that the maximum value of errors in $x(t)$ and $y(t)$ are 0.6551in and 0.6764in, respectively. Furthermore, the RMS errors of $x(t)$ and $y(t)$ are 0.2730in and 0.3657in, respectively. According to the computer simulation results, position errors in $z(t)$ and orientation errors are negligible. As shown in Figure 21, the maximum velocity errors in $x(t)$ and $y(t)$ are -5.8144in/sec and 6.1431in/sec. It also shows that the RMS errors of velocities in $x(t)$ and $y(t)$ are 0.8217 in/sec and 0.7790 in/sec, respectively. According to Figure 22 which presents the actual and desired circular paths, the tracking quality is extraordinary in spite of the step changes in payload.

8 Conclusion

In this report, we have considered the kinematic analysis and control of a 7 DOF kinematically redundant manipulator which is the slave arm of a dual-arm telerobot testbed developed at GSFC to investigate the feasibility of telerobotic applications in space. The forward kinematic transformation for the slave arm was derived and simplified for real-time implementation. Employing the method of vector cross product, we obtained the slave arm Jacobian matrix and computed its inverse using the Moore-Penrose pseudo-inverse method. The concept of quaternion was reviewed for representing the orientation of the slave arm end-effector and the relationship between quaternion and differential rotations was established. Computer simulation was performed to verify the efficiency of the Jacobian in converting joint velocities to Cartesian velocities and to investigate the accuracy of the Jacobian pseudo-inverse. The equivalence between differential rotations and quaternion was also verified through computer simulation. Simulation results showed that the maximum sampling time which ensures the efficiency of the Jacobian, its pseudo-inverse, and the quaternion representation was about 10 msec. Three control schemes were proposed for controlling the compliant and non-compliant motion of the slave arm and simulation study results were presented and discussed. The development of the Cartesian adaptive control scheme was presented and computer simulation was performed to evaluate the tracking performance of the developed control scheme. Current research activities are focusing on the implementation of the developed mathematical results and proposed control schemes for real-time control applications.

References

- [1] Klein, C.A., and Huang, C.H., "Review of Pseudoinverse Control for Use with Kinetically Redundant Manipulators," *IEEE Trans. Sys., Man, and Cyber.*, pp. 245-250, March 1983.
- [2] Hanafusa, H., Yoshikawa, T., and Nakamura, Y., "Analysis and Control of Articulated Robot Arms with Redundancy," *Proc. 8th IFAC Triennial World Congress*, Kyoto, Japan, pp. 1927-1932, 1981.
- [3] Yoshikawa, T., "Analysis and Control of Robot Manipulators with Redundancy," *Proc. 1st Intern. Symp. on Robotics Research*, New Hampshire, pp. 735-747, 1983.
- [4] Baillieul, J., Hollerbach, J., and Brockett, R., "Programming and Control of Kinetically Redundant Manipulators," *Proc. 23rd IEEE Conf. on Decision and Control*, pp. 768-774, 1984.
- [5] Nguyen, C.C., Zhou, Z.L., and Mosier, G.E., "Lyapunov-Based Direct Adaptive Control of Kinetically Redundant Telerobot Manipulators," *Proc. IASTED International Symposium on Adaptive and Knowledge-Based Control and Signal Processing*, Honolulu, Hawaii, pp. 14-18, August 1989.

- [6] **Nguyen, C.C., Zhou, Z.L., and Mosier, G.E.**, "Joint-Space Adaptive Control of a Redundant Telerobot Manipulator," *Proc. 4th IEEE International Symposium on Intelligent Controls*, Albany, New York, pp. 59-65, September 1989.
- [7] **Nguyen, C.C., Zhou, Z.L., and Mosier, G.E.**, "A Computationally Efficient Error-Based Adaptive Control Scheme for Kinematically Redundant Manipulators," in *Robotics and Manufacturing: Recent Trends in Research, Education and Applications*, Vol. 3, edited by M. Jamshidi and M. Saif, ASME Press, New York, pp. 473-480, July 1990.
- [8] **Nguyen, C.C., Zhou, Z.L., and Mosier, G.E.**, "Compliant Control of a Redundant Telerobot Manipulator via a Position/Force Control Scheme," in *Robotics and Manufacturing: Recent Trends in Research, Education and Applications*, Vol. 3, edited by M. Jamshidi and M. Saif, ASME Press, New York, pp. 505-512, July 1990.
- [9] **Schnurr, R., O'brien, M., and Cofer, S.**, "The Goddard Space Flight Center (GSFC) Robotics Technology Testbed," *NASA Conference on Space Telerobotics*, Pasadena, January 1989.
- [10] **Craig, J.J.**, Introduction to Robotics, 2nd Edition, *Addison-Wesley, Inc.*, Reading, Ma, 1989.
- [11] **Fu, K.S. et.al.**, Robotics: Control, Sensing, Vision, and Intelligence, *McGraw Hill*, New York 1987.
- [12] **Raibert, M.H. and Craig, J.J.**, "Hybrid Position/Force Control of Manipulators," *Trans. ASME, Journal of Dynamic Systems, Measurement, and Control*, Vol. 102, pp. 126-133, 1981.
- [13] **Yuan, J.S.C.**, "Closed-Loop Manipulator Control Using Quaternion Feedback," *IEEE Journal of Robotics and Automation*, Vol. 4., No. 4, pp. 434-440, August 1988.
- [14] **Burdick J. and Seraji, H.**, "Characterization and Control of Self-Motions in Redundant Manipulators," *Proc. NASA Conf. on Space Telerobotics*, Pasadena, CA, 1989.
- [15] **Paul, R.P.**, Robot Manipulators: Mathematics, Programming, and Control, *MIT Press*, 1981.
- [16] **Seraji, H.**, "Configuration Control of Redundant Manipulators: Theory and Implementation," *IEEE Transactions of Robotics and Automation*, Vol. 5, No. 4, pp. 472-490, 1989.
- [17] **Colbaugh, R.D.**, "Adaptive Position and Force Control of Redundant Robot Manipulators," in *Robotics and Manufacturing: Recent Trends in Research, Education, and Application*, edited by M. Jamshidi et al, ASME Press, New York, pp. 319-328, 1988.
- [18] **Egeland, O.**, "Cartesian Control of a Hydraulic Redundant Manipulator," *Proc. IEEE Intern. Conf. on Robotics and Automation*, Raleigh, pp. 1081-1087, April 1987.
- [19] **Hsu, P., Hauser, J., Sastry, S.**, "Dynamic Control of Redundant Manipulators," *Proc. IEEE Intern. Conf. on Robotics and Automation*, pp. 1081-1087, Philadelphia, 1988.

- [20] **Kazerounian, K., Wang, S.**, "Global versus Local Optimization in Redundancy Resolution of Robotic Manipulators," *International Journal of Robotics Research* Vol. 7, No. 5, pp. 3-12, 1988.
- [21] **Nguyen, C.C., Pooran, F.J.**, "Joint-Space Adaptive Control of Robot End-Effectors Performing Slow and Precise Motions," *Proc. 21st Southeastern Symposium on System Theory*, Florida, pp. 547-552, March 1989.
- [22] **Nguyen, C.C., Pooran, F.J.**, "Adaptive Force/Position Control of Robot Manipulators with Closed-Kinematic Chain Mechanism," in *Robotics and Manufacturing: Recent Trends in Research, Education, and Application*, edited by M. Jamshidi et al, ASME Press, New York, pp. 177-186, 1988.
- [23] **Seraji, H.**, "Direct Adaptive Control of Manipulators in Cartesian Space," *Journal of Robotic Systems*, Vol. 4, No. 1, pp. 157-178, 1987.
- [24] **Seraji, H.**, "Configuration Control of Redundant Manipulators: Theory and Implementation," *IEEE Transactions on Robotics and Automation*, pp. 472-490, August 1989.
- [25] **Nguyen, C.C., Zhou, Z.L., Mosier, G.E.**, "Joint-Space Adaptive Control of a Redundant Telerobot Manipulator," *Proc. Fourth IEEE International Symposium on Intelligent Control*, Albany, New York, pp. 59-65, September 1989.
- [26] **Nguyen, C.C., Mosier, G.E.**, "Model Reference Adaptive Control of a Telerobot System," *Proc. ISMM Intern. Symp. Computer Applications in Design, Simulation and Analysis*, Nevada, pp. 282-285, Feb. 1989.
- [27] **Colbaugh, R. and Glass, K.** "Cartesian Control of Redundant Manipulators," *Journal of Robotic Systems*. Vol. 6, No. 4, pp. 427-459, 1989.
- [28] **Colbaugh, R., Seraji, H., and Glass, K.** "Obstacle Avoidance for Redundant Robots Using Configuration Control," *Journal of Robotic Systems*. Vol. 6, No. 6, pp. 721-744, 1989.
- [29] **Nguyen, C.C., Zhou, Z.L., Mosier, G.E.**, "Kinematic Analysis and Control of A 7-DOF Redundant Telerobot Manipulator," *Proc. 22nd Southeastern Symposium of System Theory*, Cookeville, Tennessee, pp. 71-77, March 1990.
- [30] **Landau, Y.D.**, *Adaptive Control: The Model Reference Approach*, Marcell Dekker, 1979.
- [31] **Craig, J.J.**, Introduction to Robotics, 2nd Edition, *Addison-Wesley, Inc.*, Reading, Ma. 1989.
- [32] **Chen, R., Ou, Y.J.**, "Dynamic Formulation for Efficient Digital Simulation of Telerobotic Manipulation," *Final Report, Grant No. NAG 5-1019*, NASA/GSFC, December 1988.
- [33] **Nemec, B., Zlajpah, L., Matko, D.**, "Adaptive Control of Robots Using the Kalman Filter," *Int. Journal of Robotics and Automation*, Vol. 4, No.1, pp. 19-26, 1989.



Figure 1: The GSFC Telerobot Testbed

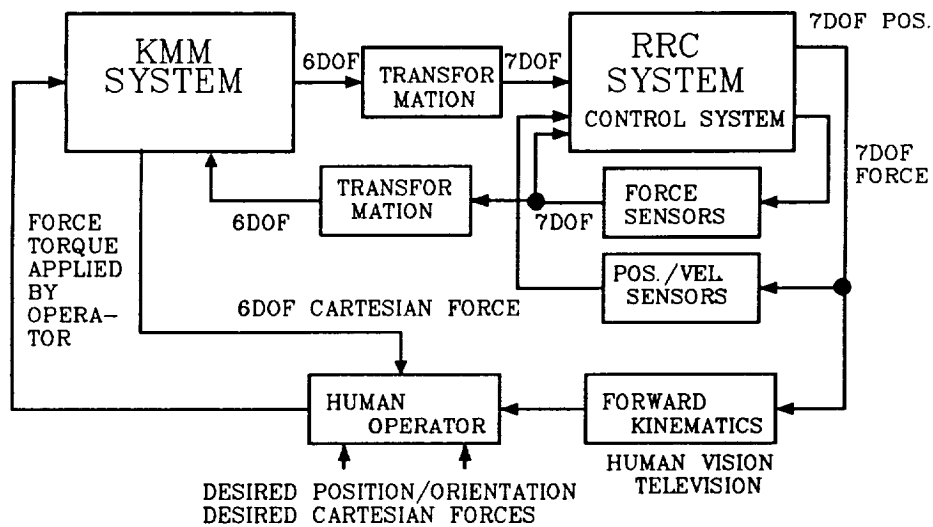


Figure 2: Block diagram of the GSFC Telerobot Testbed

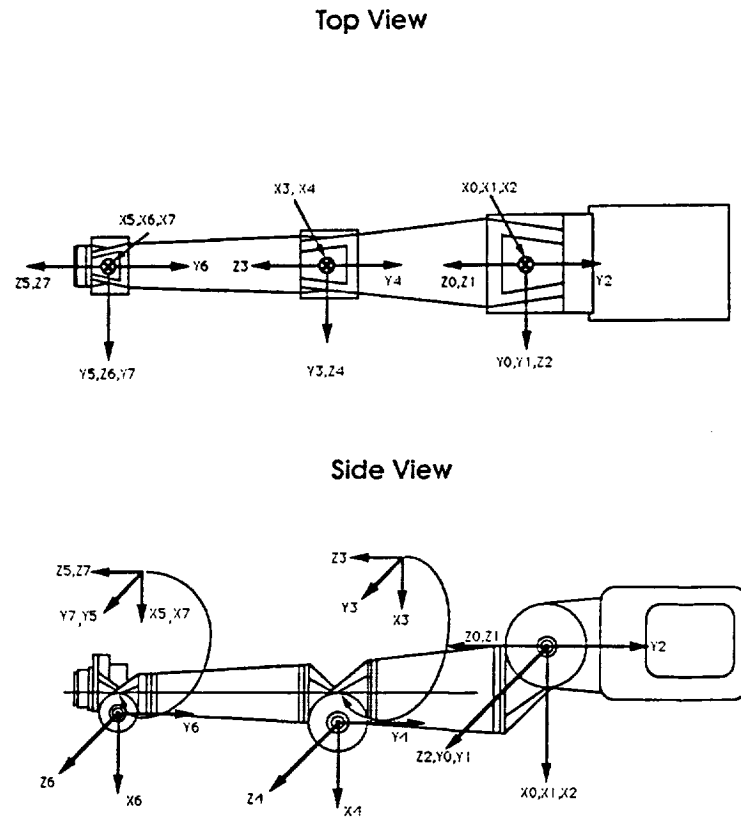


Figure 3: Coordiate frame assignment for the RRC K-1607 slave arm

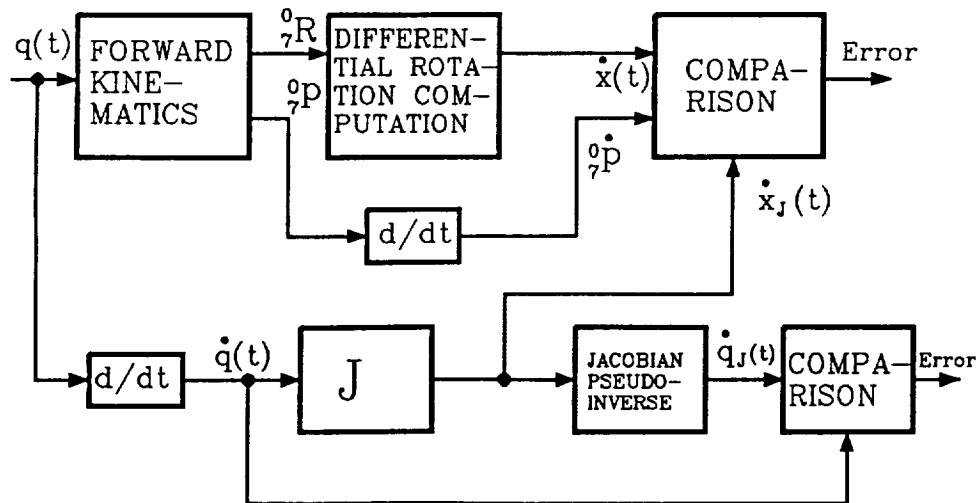


Figure 4: Computer simulation scheme for Part 1 and Part 2

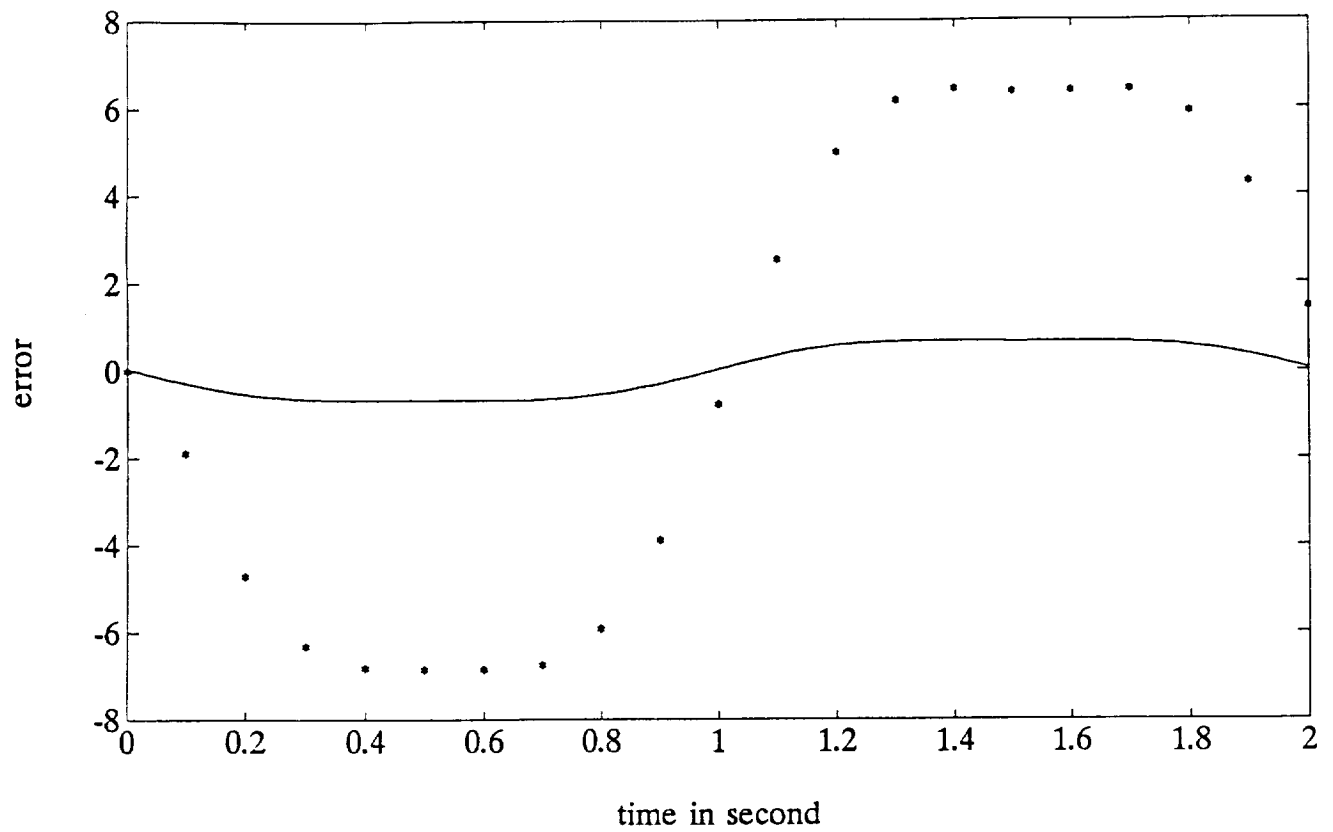


Figure 5: Errors of x-axis velocities

Sampling times: 10 msec (solid line), 100msec (dotted line)

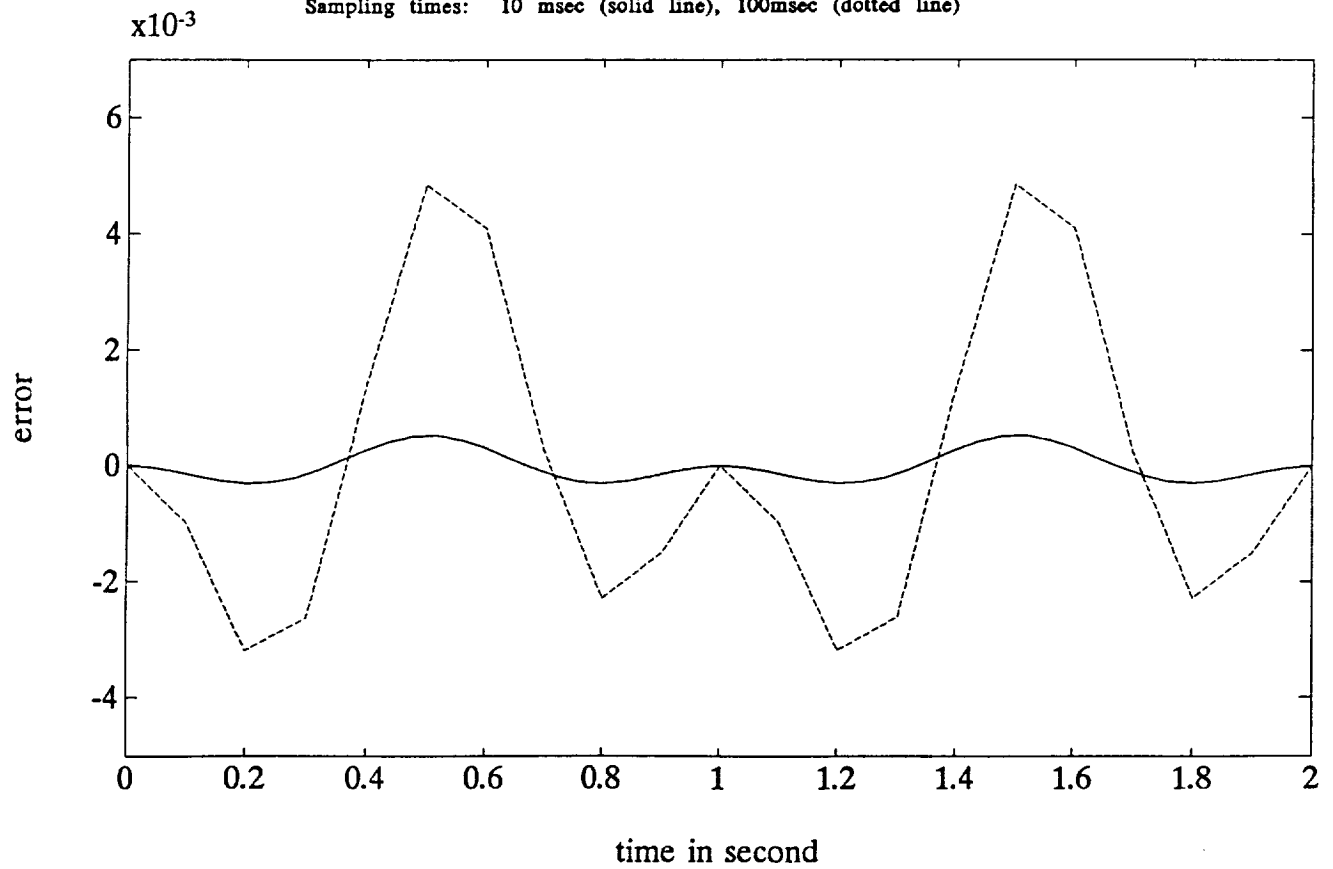


Figure 6: Errors of x-axis angular velocities

Sampling times: 10 msec (solid line), 100msec (dotted line)

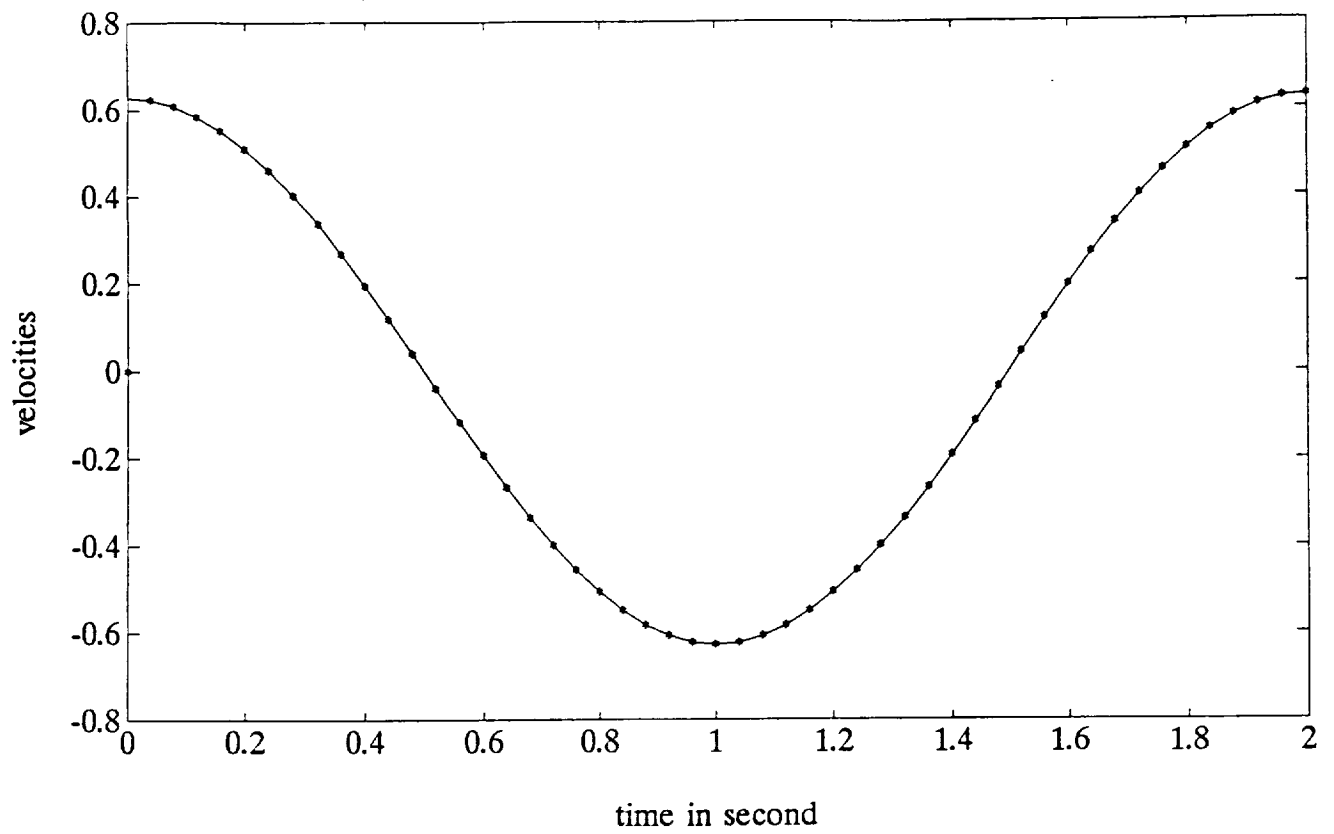


Figure 7: Velocities of joint angle 1 for sampling time of 10msec

$\dot{\theta}_{1j}$ (asteric line), $\dot{\theta}_1$ (dotted line)

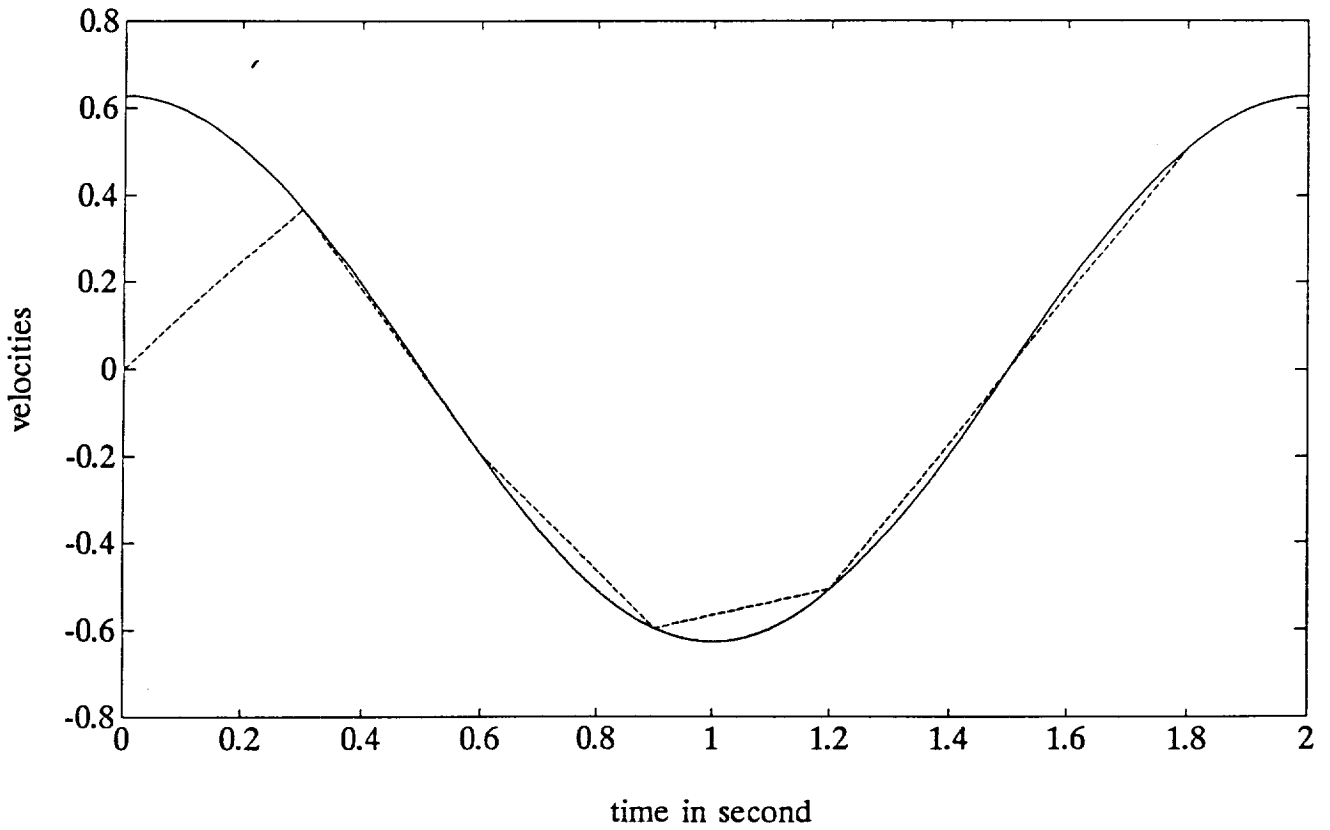


Figure 8: Velocities of joint angle 1 for sampling time of 100msec

$\dot{\theta}_{1j}$ (asteric line), $\dot{\theta}_1$ (dotted line)

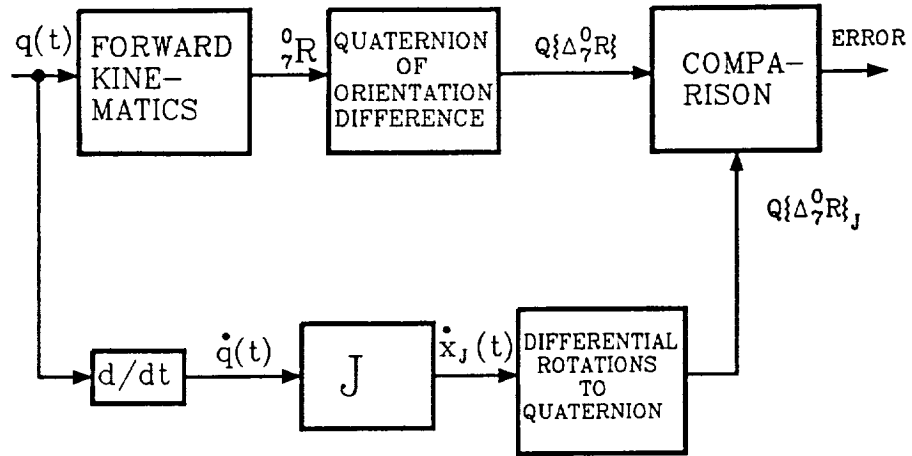


Figure 9: Computer simulation scheme for Part 3

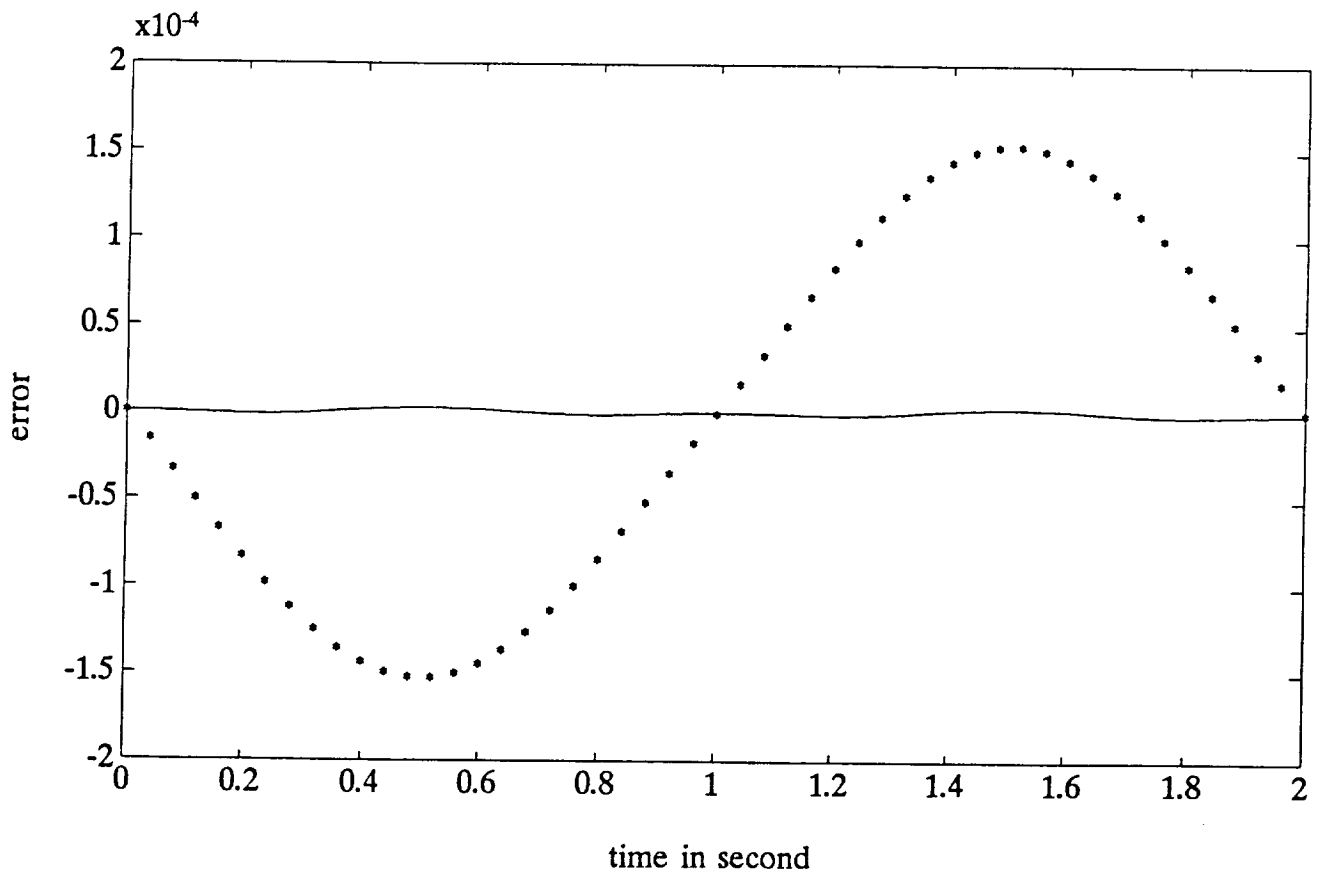


Figure 10: Quaternion errors for sampling time of 10 msec
 $\delta\beta$ (solid line), $\delta\gamma$ (asterisk line)

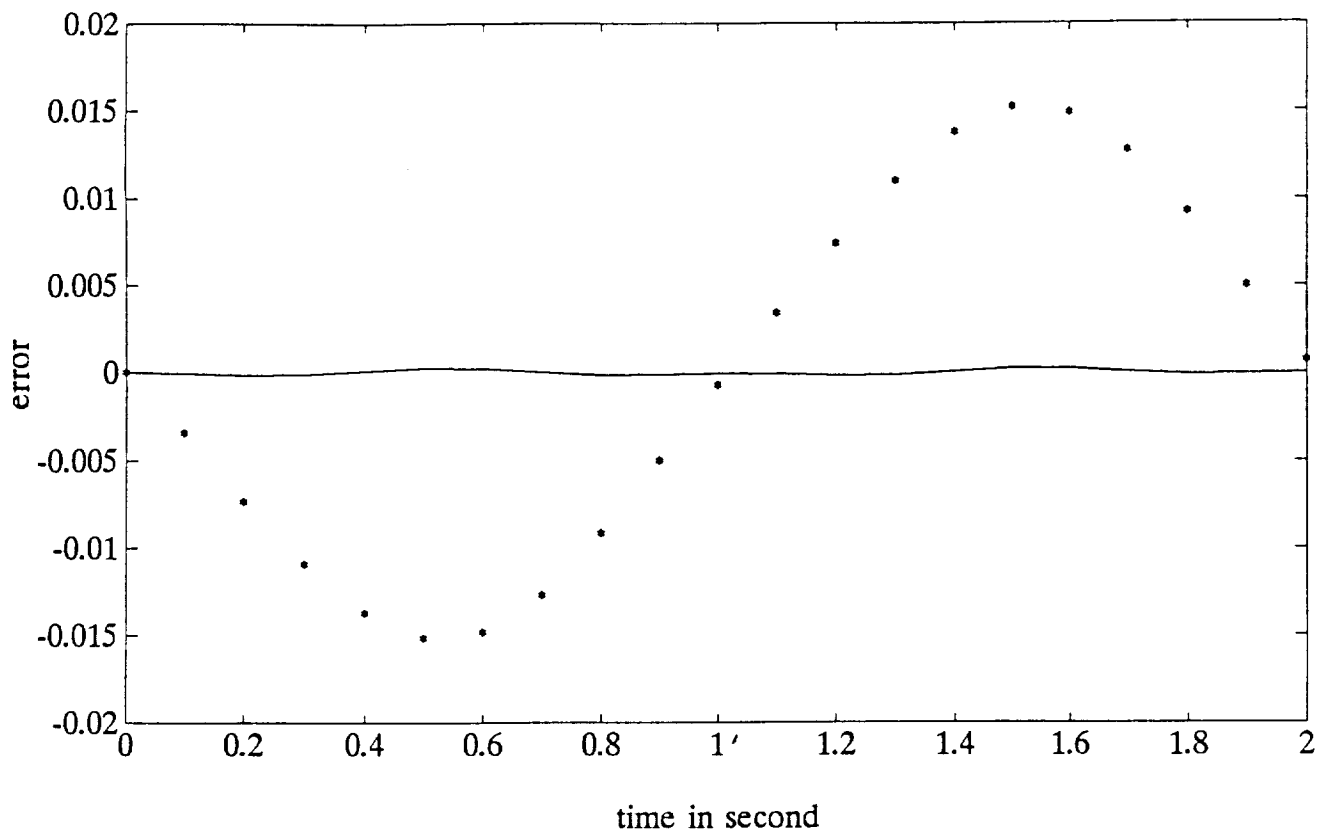


Figure 11: Quaternion errors for sampling time of 100 msec
 $\delta\beta$ (solid line), $\delta\gamma$ (asterisk line)

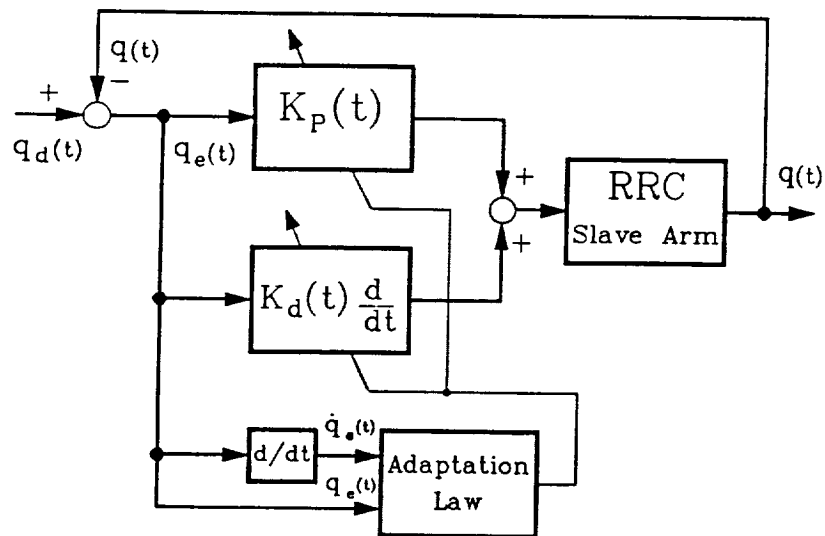


Figure 12: The joint-space adaptive control scheme

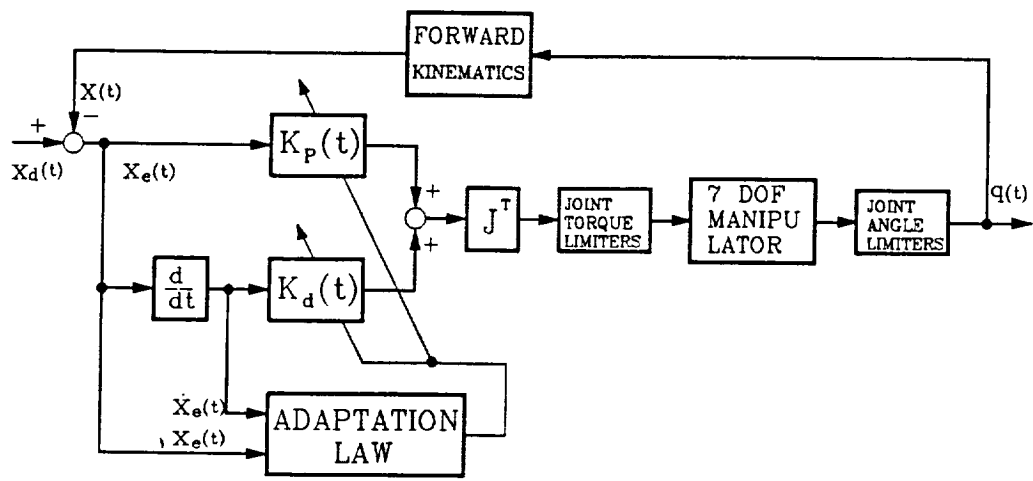


Figure 13: The Cartesian-space adaptive control scheme

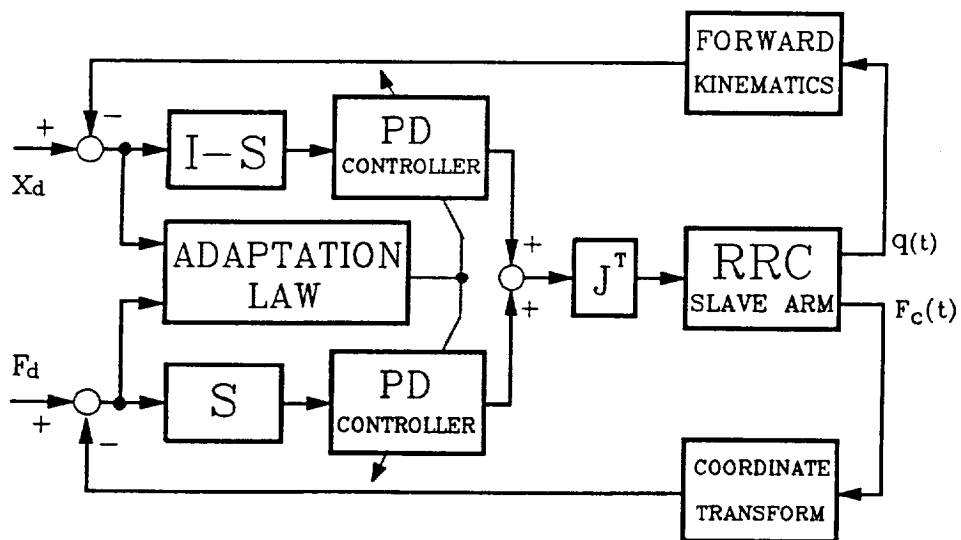


Figure 14: The hybrid adaptive control scheme

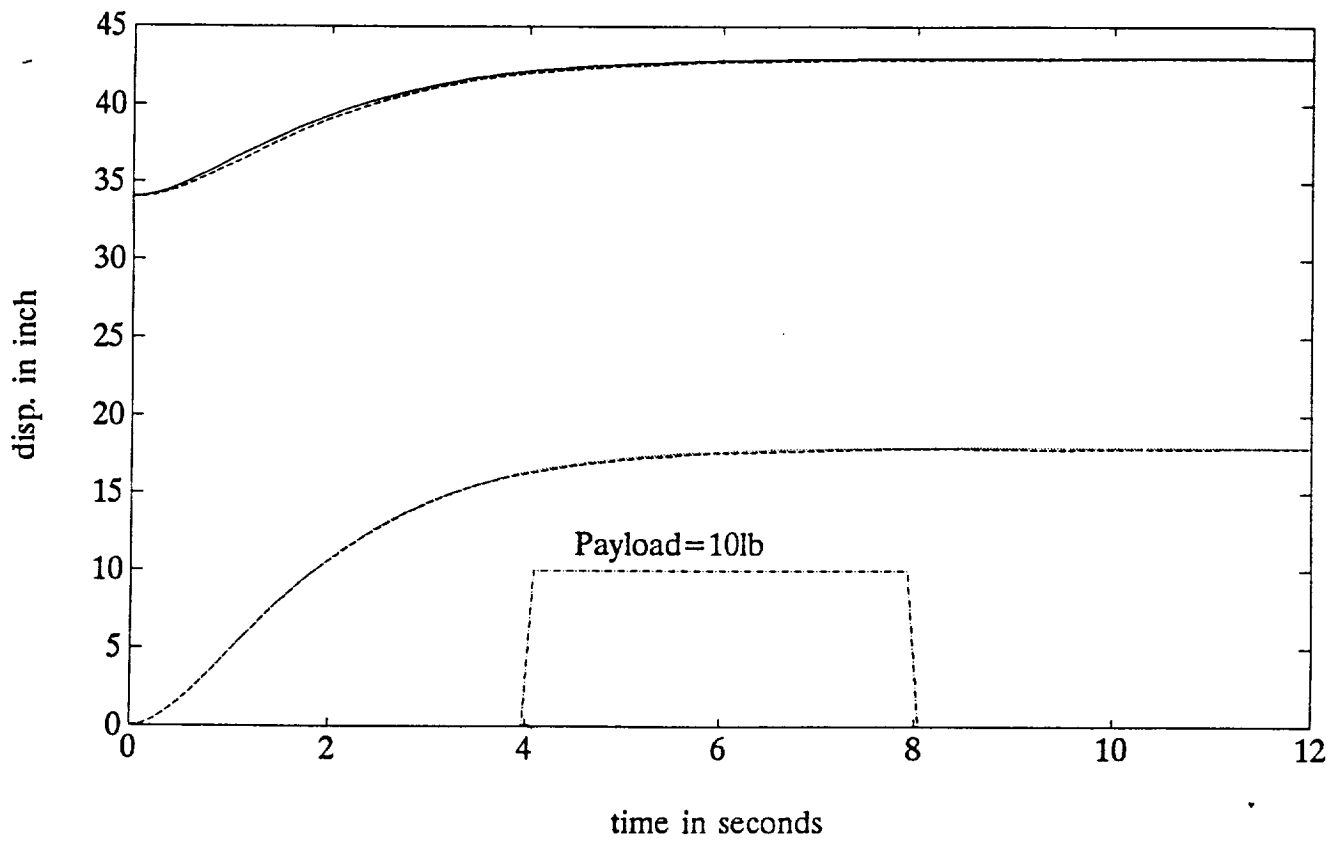


Figure 15: Time histories of $x(t)$ and $y(t)$ in tracking a straight line
 — = desired $x(t)$; .. = desired $y(t)$; -- = actual $y(t)$ and $x(t)$

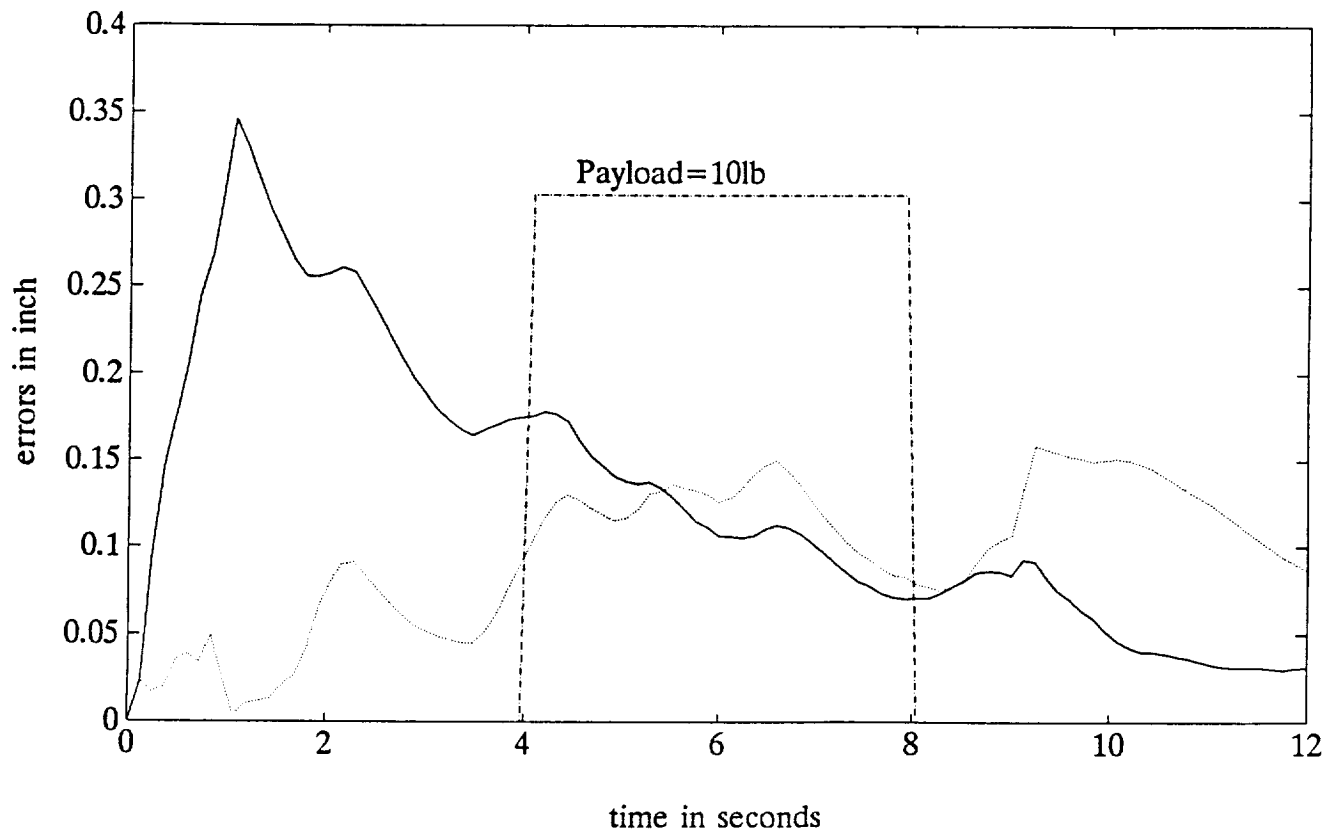


Figure 16: Position errors in tracking a straight line
 — = error in $x(t)$; .. = error in $y(t)$;

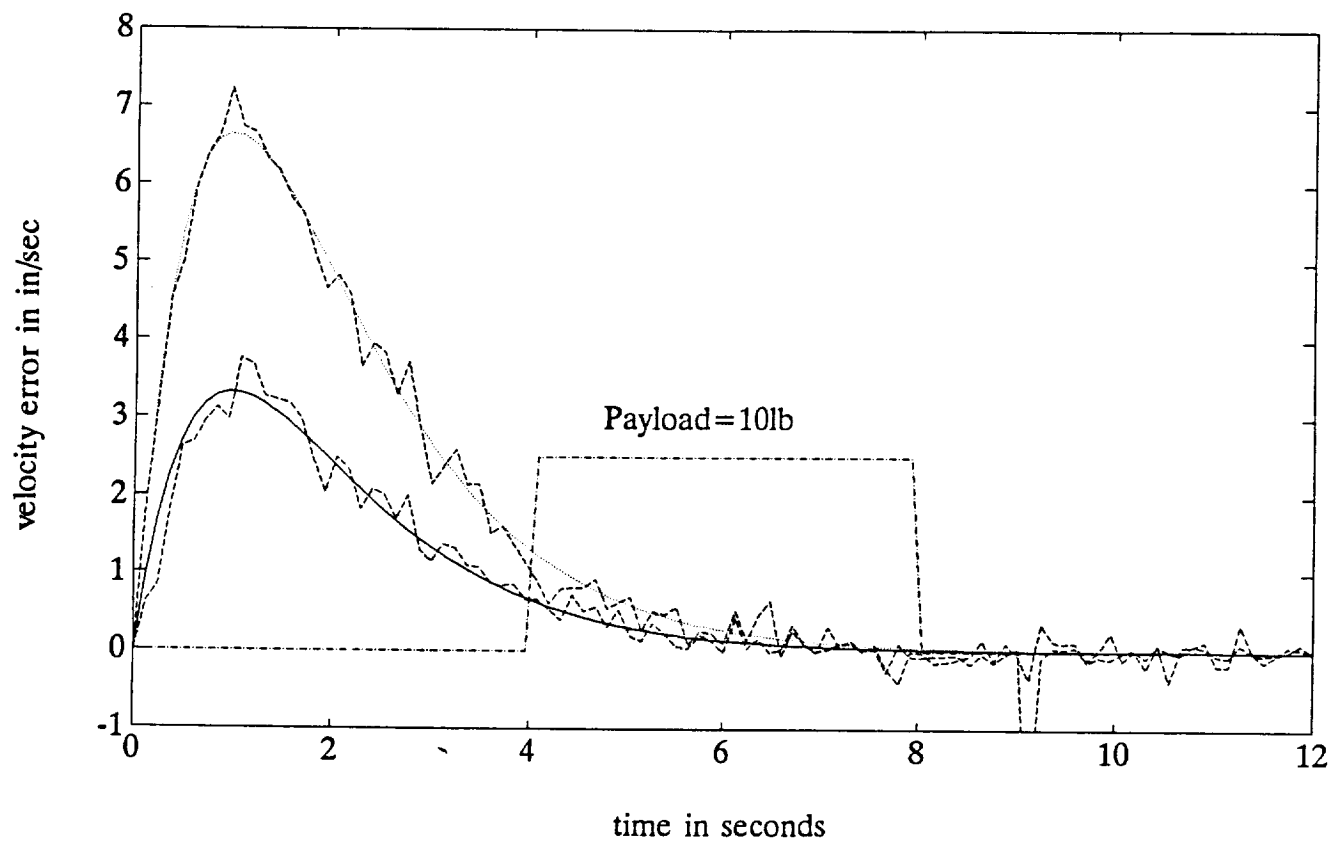


Figure 17: Velocity errors in tracking a straight line

— = error in $\dot{x}(t)$; .. = error in $\dot{y}(t)$;

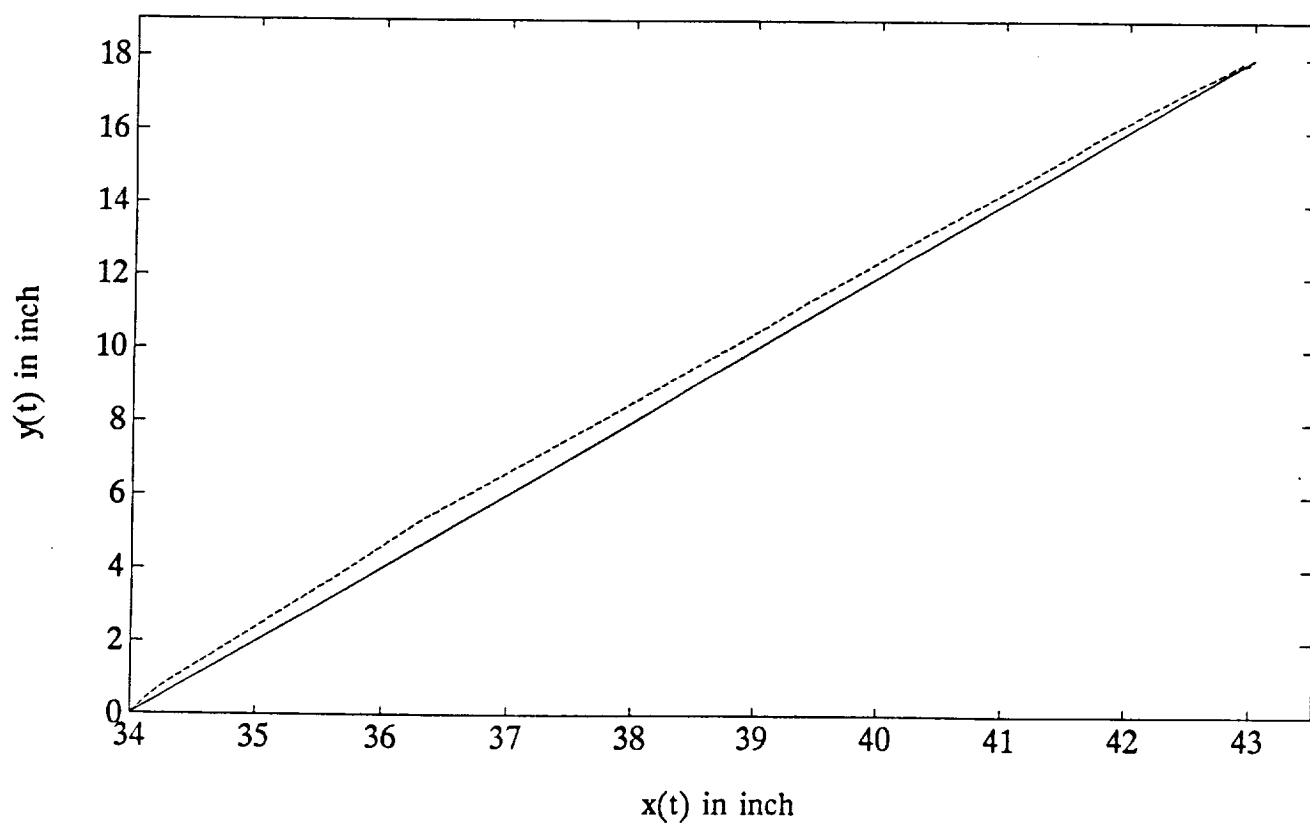


Figure 18: Tracking a straight line

— = desired path; .. = actual path

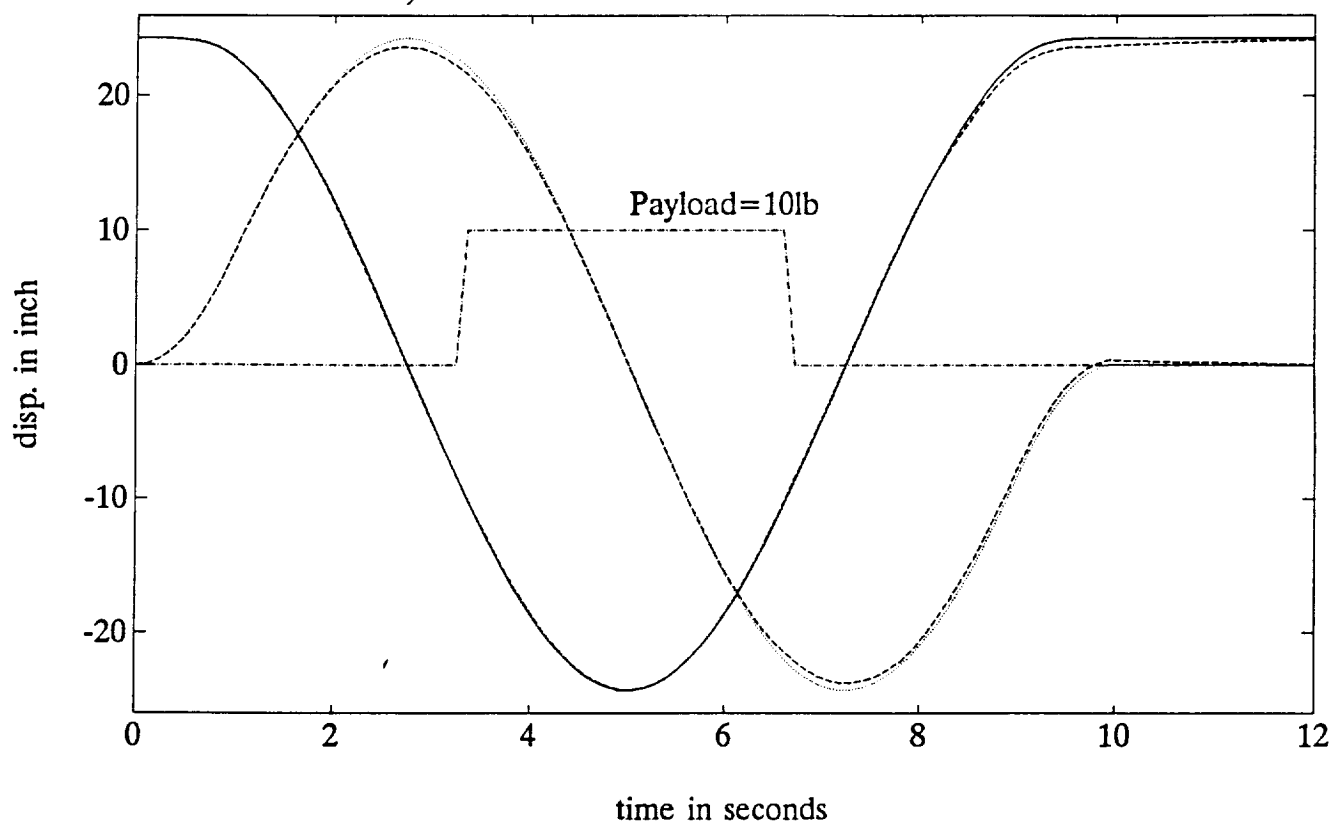


Figure 19: Time histories of $x(t)$ and $y(t)$ in tracking a circle

— = desired $x(t)$; .. = desired $y(t)$; -·- = actual $y(t)$ and $x(t)$

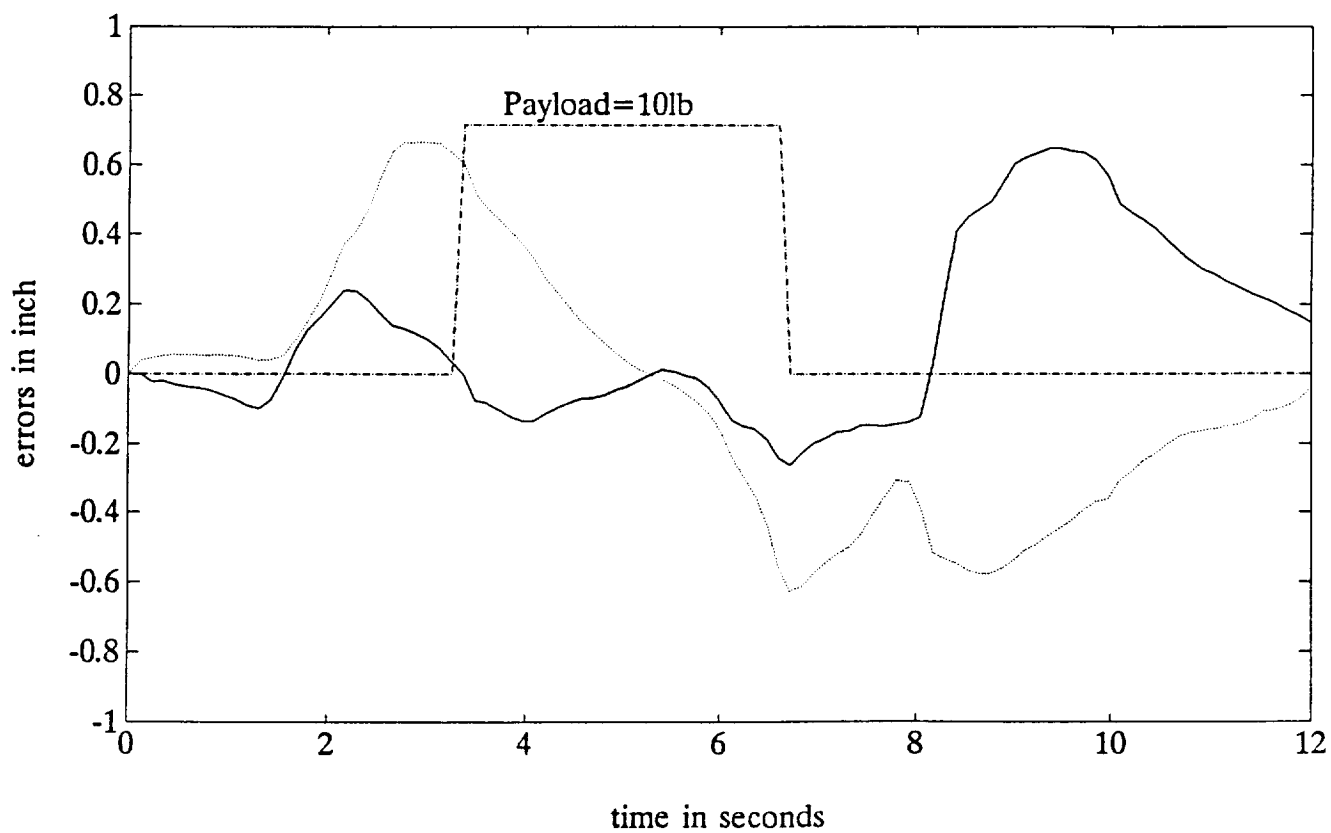


Figure 20: Position errors in tracking a circle

— = error in $x(t)$; .. = error in $y(t)$;

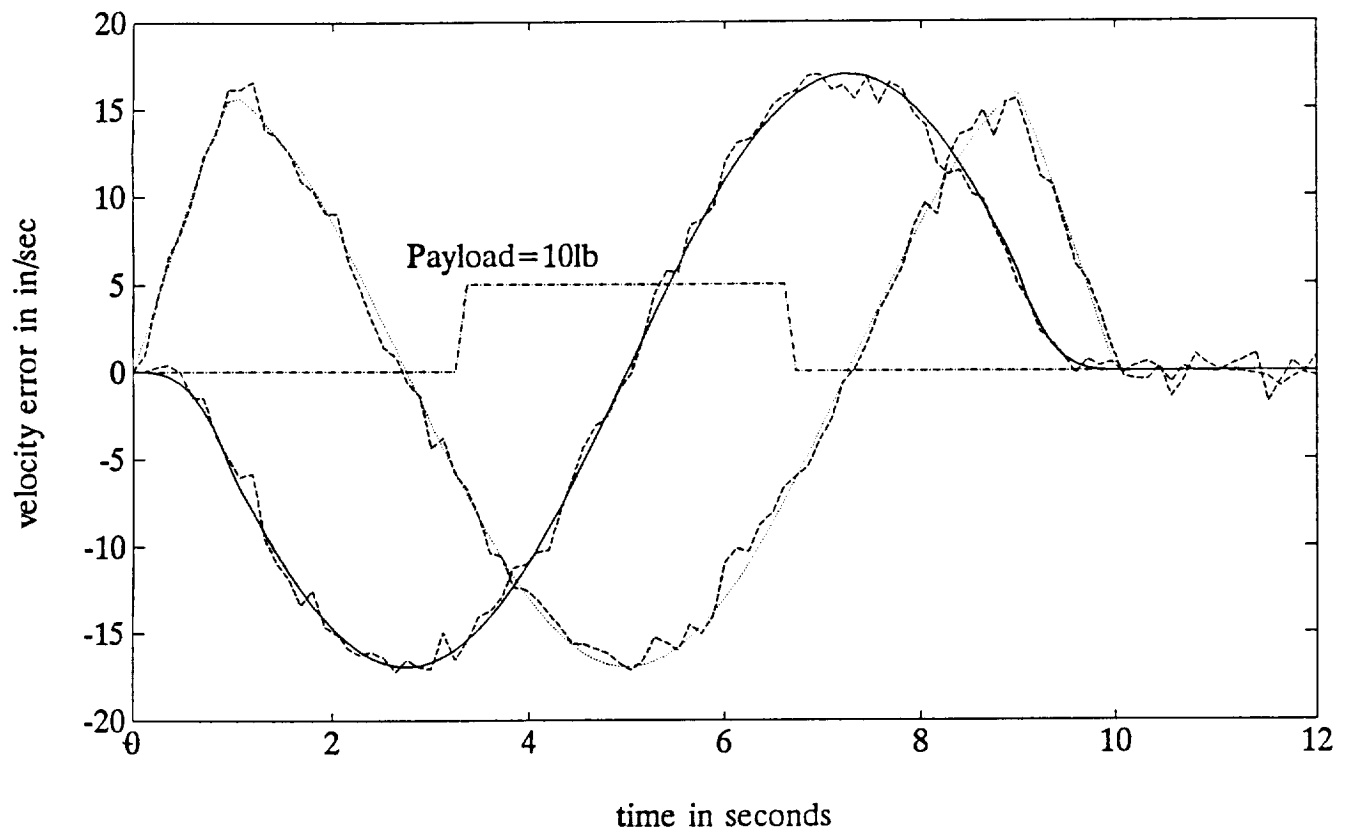


Figure 21: Velocity errors in tracking a circle
 — = error in $\dot{x}(t)$; .. = error in $\dot{y}(t)$;

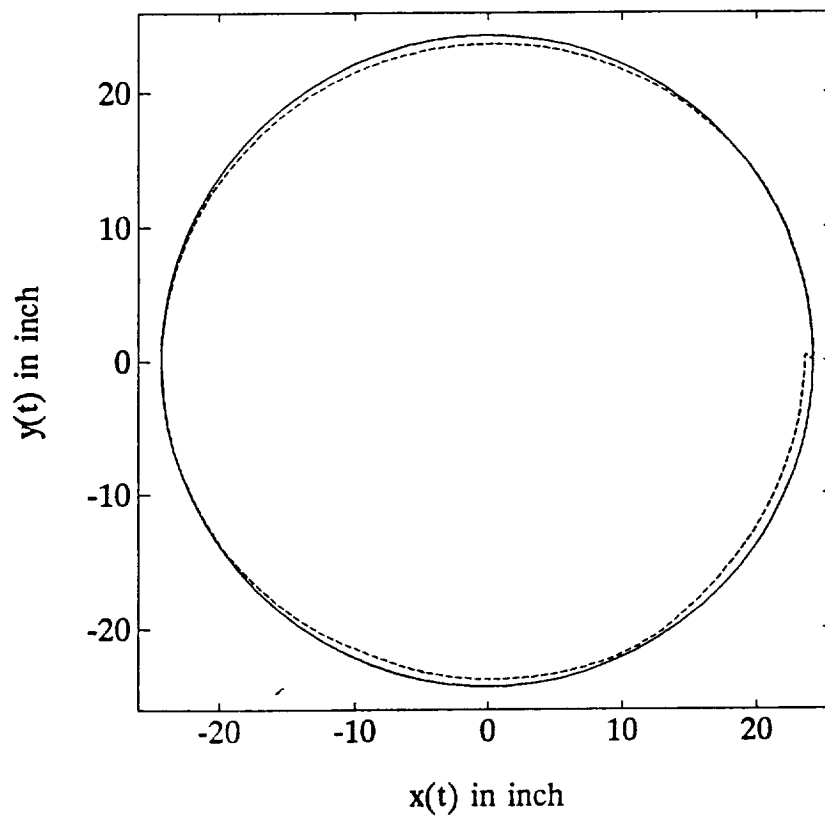


Figure 22: Tracking a circle
 — = desired path; .. = actual path